GUÍA DEL DERECHO Y EL SOFTWARE DE FUENTES ABIERTAS







Coordinación y edición: CREA'T
Autor: Malcolm Bain, id law partners.
Maquetación interactiva: www.linea4.eu
Agradecimientos por los comentarios y contribuciones realizados por los socios de id law partners (Manuel Martínez Ribas en particular) y por los proyectos y empresas mencionados en el texto: UOC / Proyecto Campus, UPC/Càtedra de Programari Lliure, Openbravo SL, openTrends Soluciones y Sistemas SL, Zylk.net SL, Axiom Tech Ltd) y el CENATIC.

ÍNDICE







CAPÍTULO 8: OTRAS NORMAS APLICABLES 126





CAPÍTULO 1

INTRODUCCIÓN

1.1. OBJETIVOS DE LA GUÍA

A medida que el Software de Fuentes Abiertas (**SFA**), conocido también en inglés como Open Source Software, adquiere más importancia en todos los sectores de la economía y en la vida personal, se hace más necesario entender cuáles son los derechos y obligaciones de las personas y organizaciones frente al mismo.

Esta guía pretende presentar y comentar, brevemente y con un enfoque generalista, los principales aspectos jurídicos del SFA. Se trata de un documento divulgativo. Nuestro fin no es redactar un tratado sobre la protección jurídica y la explotación del software distribuido bajo licencias de SFA, sino presentar una guía práctica y accesible a cualquier persona u organización.

Dirigimos esta guía de manera general a las diversas personas, empresas y entidades en todos los sectores que participan la cadena de creación, integración y uso de software de fuentes abiertas. En particular, se dirige a:

- Profesionales en los departamentos de sistemas y de tecnología de las organizaciones usuarias de software, tanto en el sector privado (la empresa) como el institucional o público.

- Estudiantes, investigadores o profesores universitarios, que desarrollan software en un marco institucional educativo o de manera independiente, solos o con un grupo de trabajo o investigación; con la intención de utilizar o liberar software bajo licencia de fuentes abiertas.

- Desarrolladores que participan en proyectos o comunidades ya existentes de software de fuentes abiertas, que suponen la participación de varios actores en la creación y divulgación del software a lo largo del tiempo; así como las personas responsables de los mismos; en particular los responsables de la gestión del código y de la comunidad.

- Consultores informáticos (una empresa, un profesional) que proveen soluciones informáticas como respuesta a las necesidades particulares de sus clientes, proporcionando software de nueva creación o integrando y personalizando software y soluciones informáticas de fuentes abiertas de terceros.

- Gestores y "propietarios" de proyectos en estas mismas organizaciones que gestionan la adquisición e implementación de proyectos informáticos.





- Usuarios particulares, que deseen conocer más sobre los derechos y libertades del software de fuentes abiertas y sus aspectos jurídicos.

Nuestro objetivo es ayudarles a entender cuáles son los derechos y obligaciones que se derivan de las licencias y el derecho aplicable al software de fuentes abiertas: incentivar su desarrollo, adquisición, uso y liberación; asegurar que sus beneficios son explotados correctamente; y eliminar cualquier reticencia que pueda existir por desconocimiento de las cuestiones jurídicas.

Advertencia

Esta guía se centra en el derecho español, si bien, debido a los esfuerzos de armonización internacional, encontraremos similitudes entre lo aquí expuesto y las normas jurídicas existentes en distintos países de la Unión Europea e incluso a nivel mundial. Pero no por ello debemos obviar ciertas diferencias de localización que pueden hacer que una misma controversia se resuelva jurídicamente de una u otra manera en los diferentes paises en los que se plantee.

1.2. EL SOFTWARE DE FUENTES ABIERTAS

Antes de entrar en temas propiamente jurídicos, introduciremos algunos conceptos que se van a repetir a lo largo de esta Guía. En concreto, queremos definir y aclarar tres conceptos que están relacionados entre sí: **software de fuentes abiertas, software libre y copyleft.**

¿Qué es el Software de Fuentes Abiertas (SFA)? Básicamente, es software que se distribuye bajo una licencia de fuentes abiertas. Entonces ¿qué es una licencia de SFA? Es una licencia que garantiza las "libertades del software" y cumple con las directrices establecidas por la Iniciativa para el Software de Fuentes Abiertas (Open Source Initiative u OSI[1]), tal como explicamos a continuación.

1.2.1. Las libertades del software

El principio general que subyace al SFA, así como el concepto paralelo de software libre, es el concepto de libertad. Su sentido desde una perspectiva legal es claro: que carece de prohibiciones.

Pero el marco jurídico del software es por defecto restrictivo. Las leyes reservan automáticamente los derechos de explotación a los autores y titulares del software. Para usar y explotar un software, el usuario necesita la autorización del titular, habitualmente otorgada por un contrato llamado licencia. Tradicionalmente, las licencias de software son restrictivas (licencias propietarias o privativas[2]), reservando la mayor parte de derechos al titular y limitando los que se ceden al usuario. Con esto, el titular garantiza su monopolio

1 En www.opensource.org

2 Utilizamos los adjetivos privativo y propietario para referirnos al software bajo licencias restrictivas. Sería más correcto referirnos a licencias no libres o cerradas, pero no es una terminología habitual en el sector.





(asegurando sus ingresos en base a la venta de copias del software) y, en cierta manera, mantiene cautivo al usuario.

En contraste, las licencias de software libre permiten a los usuarios utilizar un programa de ordenador[3] libremente, disminuyendo sustancialmente esta cautividad. **Las libertades que los desarrolladores de software y titulares de los derechos deben conceder a los usuarios se suelen expresar de la siguiente manera[4]:**

- **La libertad de usar el programa, con cualquier propósito (libertad 0).**
- **La libertad de estudiar cómo funciona el programa, y adaptarlo a sus necesidades (libertad 1).**
- **La libertad de distribuir copias (libertad 2).**
- **La libertad de mejorar el programa y hacer públicas las mejoras a los demás (libertad 3).**

Ciñéndonos a lo establecido por la ley española, para que una licencia garantice estas libertades debe permitir la ***reproducción, transformación, comunicación pública y distribución del programa original[5]. Para ello, el licenciatario debe tener acceso al código fuente***. De esta forma, cualquier usuario de software bajo licencia de SFA tiene derecho a:

A) Instalar y ejecutar el programa (con objetivos comerciales o de uso privado),
B) Reproducirlo,
C) Modificarlo
D) Redistribuirlo y comunicarlo públicamente.

1.2.2. El software de fuentes abiertas

La expresión **"software de fuentes abiertas"** es una traducción de "Open Source Software" y se refiere a aquellos programas que se distribuyen bajo una licencia que ofrece estas cuatro libertades y cumple con las directrices de la OSI, agrupadas en la **"Definición de Software de Fuentes Abiertas"** (Open Source Definition u OSD[6]).
Estas directrices aseguran, entre otras cosas, que una licencia certificada como "de fuentes abiertas" concede a los usuarios los mencionados derechos de explotación del software, garantizando que ello se hace sin discriminación y que los licenciatarios directos pueden acceder al código fuente.

Los 10 criterios de la Open Source Definition

Los 10 criterios que debe satisfacer una licencia para que pueda considerarse de fuentes abiertas son:

1 **Redistribución libre.** La licencia no deberá impedir a nadie vender u ofrecer el software como un componente de una distribución de software compuesto, conteniendo programas de fuentes distintas. La licencia no deberá requerir el pago de regalías u otra tasa por dicha venta.

3 *En esta Guía, salvo mención expresa, usaremos los términos software y programa de ordenador de manera sinónima.*
4 *Originalmente anunciadas por la Fundación para el Software Libre (FSF - Free Software Foundation) en: La Definición de Software Libre, disponible en http://www.gnu.org/philosophy/free-sw.es.html. Dentro del movimiento del software libre y de fuentes abiertas hay diferentes concepciones de esta libertad, sin embargo todas coinciden en esta definición.*
5 *Estos conceptos se desarrollan en el Capítulo 7, sobre el Derecho de la Propiedad Intelectual.*
6 *Comentado en él capítulo 3, disponible en www.opensource.org/docs/definition.php.*





2 **Código fuente.** El programa debe incluir su código fuente y la licencia debe permitir su distribución tanto en forma de código fuente como compilada. Si no se distribuye un producto junto con su código fuente, éste debe poder obtenerse a través de un medio accesible y claramente publicado, por no más que un coste razonable de reproducción (preferentemente una descarga a través de Internet, sin cargo). El código fuente se considera la forma preferida del código por parte de un desarrollador para modificarlo y no está permitido ofuscarlo deliberadamente. Las formas intermedias, tales como la salida de un pre-procesador o traductor, tampoco están permitidas.

3 **Obras derivadas.** La licencia tiene que permitir la realización de modificaciones y obras derivadas, así como su distribución bajo los mismos términos de la licencia del software original.

4 **Integridad del código fuente del autor.** En el caso de haberse modificado el código fuente, la licencia sólo podrá prohibir su distribución si permite suministrar archivos parche junto con el mismo, con el objetivo de modificar el programa en el momento de la compilación. La licencia debe permitir, explícitamente, la distribución de software creado a partir del código fuente modificado y puede requerir que las obras derivadas tengan un nombre o un número de versión distinto al del software original.

5 **La no discriminación con respecto a las personas o grupos.** La licencia no debe discriminar a ninguna persona o grupo de personas.

6 **La no discriminación con respecto a los sectores de actividad.** La licencia no debe limitar el uso del programa en ningún sector de actividad. Por ejemplo, no puede impedir que el programa sea usado en un negocio determinado o para la investigación genética, por ejemplo.

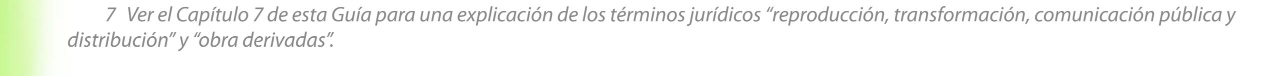
7 **Distribución de la licencia.** Los derechos asociados con el programa deben cederse a todos aquellos que lo reciben, sin que éstos tengan que firmar una licencia adicional.

8 **La licencia no tiene que ser específica de un producto.** Los derechos asociados al programa no deben depender de que éste forme parte de un producto o versión particular de software (distribución). Si un programa se extrae de una distribución, la licencia de éste debe ceder los mismos derechos que los contemplados en la distribución original.

9 **La licencia no debe limitar a otro software.** La licencia no debe imponer restricciones sobre otro software que se distribuya conjuntamente con el software licenciado. Por ejemplo, la licencia no debe exigir en que todos los otros programas distribuidos en el mismo medio tengan que ser software de código de fuentes abiertas.

10 **La licencia debe ser neutra respecto de la tecnología.** Ninguna disposición de la licencia debe depender de una tecnología o un tipo de interfaz particular.

Vemos que el criterio 1 obliga a conceder los derechos de reproducción, comunicación pública y distribución; el criterio 3, los derechos de transformación y distribución de las obras derivadas (con o sin copyleft); y el criterio 2 el acceso al código fuente[7]. Los demás criterios aseguran la neutralidad y la no discriminación en cuanto a usos, tecnologías, programas y maneras de distribuir el software.


7 Ver el Capítulo 7 de esta Guía para una explicación de los términos jurídicos "reproducción, transformación, comunicación pública y distribución" y "obra derivadas".





Las licencias usualmente conocidas como de fuentes abiertas son, en su mayoría, también licencias libres, y viceversa. Jurídicamente, son dos maneras de hablar de un mismo tipo de licencia.

Licencias libres vs. licencias de fuentes abiertas

La diferencia entre el movimiento de software de "fuentes abiertas" y el "libre" es de carácter filosófico y ético. Restringiéndonos a una perspectiva jurídica, es decir, analizando específicamente las condiciones de las licencias en cuestión, podemos decir que los términos "licencia libre" y "licencia de fuentes abiertas" son sinónimos.

A menudo, para incluir ambas ideas en un solo concepto, se usa la expresión FOSS (Free and Open Source Software) o FLOSS (Free, Libre and Open Source Software). En esta Guía, usaremos la expresión Software de Fuentes Abiertas o SFA. Dentro del esquema general de estas libertades, existen varias maneras de expresarlas jurídicamente. Esto, junto con las distintas condiciones adicionales que uno puede agregar, es la causa de que existan cerca de 70 licencias de fuentes abiertas reconocidas por la OSI, cada una con sus particularidades . En especial, suele hacerse una distinción entre las licencias de SFA con y sin copyleft, un concepto importante que merece que nos detengamos para explicarlo.

Dentro del esquema general de estas libertades, existen varias maneras de expresarlas jurídicamente. Esto, junto con las distintas condiciones adicionales que uno puede agregar, es la causa de que existan cerca de 70 licencias de fuentes abiertas reconocidas por la OSI, cada una con sus particularidades[8]. En especial, suele hacerse una distinción entre las licencias de SFA con y sin copyleft, un concepto importante que merece que nos detengamos para explicarlo.

1.2.3. El copyleft

Las licencias de SFA con copyleft van más allá de garantizar las cuatro libertades básicas del software para los licenciatarios o usuarios directos. Una licencia que concede los mencionados derechos sin imponer condiciones permite al licenciatario incluir el SFA en otro software y redistribuir el resultado bajo licencia restrictiva o "cerrada[9]", de manera que los usuarios del nuevo programa no tendrán las libertades originalmente cedidas.

Con el objetivo de asegurar que cualquier usuario del software pueda disfrutar de estas libertades en todo momento, las licencias con copyleft obligan a los licenciatarios a:

- utilizar la misma licencia libre tanto para la redistribución del software original como para cualquier modificación que se realice del mismo
- a proporcionar u ofrecer acceso al código fuente a todos los usuarios.

Esta doble condición, conocida como copyleft, prohíbe la distribución del software bajo licencia privativa.

Esto ha llevado a muchos a pensar que el software libre es software con copyleft (y, llevado al extremo, software distribuido bajo la GNU General Public License o GPL, por



8 *Ver en www.opensource.org/licenses y nuestros comentarios en el Capítulo 2 de esta Guía.*
9 *Es lo que permiten las licencias de SFA "permisivas" como las licencias Berkeley Software Distribution y MIT.*





ser ésta la licencia con copyleft más habitual).[10] Pero esta afirmación es incorrecta: las licencias de SFA con copyleft son un subconjunto particular de las licencias de SFA, algo que comentaremos en el Capítulo 2.

La importancia de este concepto es tal que se suele usar como criterio de clasificación de las licencias de SFA, distinguiéndose:

- las licencias **permisivas sin copyleft** (por ejemplo, la licencia Berkeley Software Distribution o BSD),
- las licencias **con copyleft fuerte** (la GPL)
- y las licencias **con copyleft suave o mixtas** (por ejemplo la Lesser GPL o LGPL, y la Mozilla Public License o MPL).[11]

1.3. LOS EFECTOS DEL SFA Y EL CONCEPTO DE "COMUNIDAD"

El impacto de las licencias de SFA a nivel jurídico y práctico es importante y lo comentaremos con más detalle en los siguientes capítulos de esta Guía, sobre todo en los capítulos 4, 5 y 6 (dónde presentamos los aspectos legales del uso, integración y desarrollo del SFA).

La consecuencia directa de usar software bajo una licencia de SFA -y ejercer los derechos concedidos bajo la misma- es la posibilidad de:

1 Descargar el programa libremente de Internet (habitualmente de manera gratuita).

2 Realizar una instalación para probarlo y evaluarlo.

3 Modificarlo para adaptarlo a nuestras necesidades (o contratar un desarrollador-consultor para que lo haga).

4 Implementarlo en nuestro negocio en cuantos equipos sea necesario, y actualizarlo a medida que se liberen nuevas versiones.

5 Redistribuirlo (en Internet o en formato CD/DVD, etc.) para que otros puedan beneficiarse de cualquier modificación o mejora realizada.

Todo ello sin tener que negociar una licencia con un vendedor, establecer contratos de soporte exclusivos o calcular los equipos o usuarios que van a utilizar el software en cuestión.

Entre las principales consecuencias indirectas, encontramos las siguientes:

1 **Reutilización:** el derecho de poder ejecutar, modificar y redistribuir el SFA implica que hay un nivel mucho más alto de reutilización, tanto por lo que se refiere a


10 Confusión fomentada, en cierta manera, por la propaganda contraria al movimiento de SFA, conocida bajo sus siglas inglesas FUD (Fear, Uncertainty, Doubt – Temor, Incertidumbre y Duda).


11 Ver el Capítulo 2 de esta Guía para más detalles y una explicación de los términos licencias permisivas, copyleft fuerte y copyleft suave.



componentes como a aplicaciones completas para el usuario final[12] , lo que conlleva una mayor eficiencia.

2 **Independencia:** el libre acceso al código fuente, junto con el derecho de ejecutar y modificarlo, otorga al usuario un alto nivel de independencia de su proveedor, algo que puede se aprovechar para exigir mayor calidad del servicio prestado.

3 **Colaboración:** estos mismos derechos de acceso, ejecución y modificación fomentan la creación colaborativa de software (entre desarrolladores que quizá nunca se han conocido) y la corrección de errores por parte de los usuarios.

4 **Modelos de negocio:** al no poder "vender licencias", las empresas de consultoría y desarrollo de SFA suelen basar su negocio en la venta de servicios (de selección, de integración e implementación, de soporte y mantenimiento, formación, el ofrecimiento de garantías, etc.).

5 **Comunidades:** los derechos concedidos por las licencias permiten el uso intensivo y la difusión masiva del SFA a través de las redes de Internet (sobre todo a partir de repositorios como *Sourceforge*[13] o C3 de CENATIC[14]) y fomentan la creación de comunidades alrededor de los proyectos de SFA.

Utilizaremos en esta guía el término **"comunidad"** para referirnos de forma genérica al conjunto de personas y organizaciones que participan en la creación, distribución y uso de SFA. Incluye a los programadores individuales que contribuyen código a los proyectos de SFA y asociaciones locales de usuarios de tipo "usuarios de Linux" ("LUGs" [15]), el sector universitario, proyectos importantes de SFA , empresas específicamente de SFA , y asociaciones de empresas.

Finalmente, la administración pública juega un papel cada vez más importante en el sector: el Gobierno español; entidades públicas como el CENATIC[16]; las Comunidades Autónomas y Municipios españoles; la Comisión Europea y las Naciones Unidas , ya hace tiempo que están trabajando para fomentar el uso y conocimiento del SFA a distintos niveles.

1.4. ORGANIZACIÓN DE LA GUÍA Y GLOSARIO

Esta Guía se organiza en 8 capítulos, incluyendo esta introducción.

- **En el capítulo 2** presentamos las **licencias de fuentes abiertas**, entrando en profundidad en aquellas más usadas o conocidas, así como la EUPL (European Union Public License), una licencia de reciente creación destinada a software proveniente de las administraciones públicas europeas.

12 Así como a nivel de los stacks –es decir los conjunto de subsistemas de software: sistema operativo, base de datos, servidor de aplicaciones, etc. Un ejemplo conocido es el stack "LAMP": Linux, Apache, MySQl, PHP. Ver en: http://es.wikipedia.org/wiki/LAMP.
13 Mayor repositorio de SFA en Internet, con unos 180.000 proyectos de SFA. En www.sf.net.
14 Comunidad de Conocimiento Compartido de CENATIC. En http://forja.cenatic.es
15 LUG : siglas en inglés de Linux User Groups.
16 Centro Nacional de Referencia de Aplicación de las TIC basadas en fuentes abiertas, http://www.cenatic.es





- **En el capítulo 3** dilucidamos algunas **cuestiones prácticas y jurídicas que suelen surgir a la hora de usar licencias SFA:** la selección de las licencias, su naturaleza jurídica, las licencias duales, la compatibilidad de licencias y las garantías.

- **En los capítulos 4, 5 y 6** comentamos brevemente aquellos **aspectos jurídicos que deben tener en cuenta los usuarios, integradores y desarrolladores de SFA** respectivamente, ilustrados con algunos casos de estudio o referencias.

- **En los capítulos 7 y 8** introducimos, a modo de referencia, **el marco jurídico de la protección y explotación del software** (la propiedad intelectual y la propiedad industrial), así como su aplicación al SFA.

Glosario

Esta guía es doblemente técnica, tanto desde la perspectiva jurídica como de las tecnologías objeto del estudio. Por ello, se utilizarán una serie de términos que definimos a continuación:

- **Copyleft:** condición de una licencia de SFA que obliga, en caso de redistribuir el software, a utilizar la misma licencia de SFA sin restricciones adicionales y a asegurar que el usuario destinatario tiene acceso al código fuente.

- **Copyright notices:** los avisos de propiedad o titularidad de derechos de autor.

- **Disclaimers:** cláusulas de limitación de garantías y responsabilidades.

- **Distribución:** jurídicamente, la distribución consiste en hacer accesible al público el original o copias de un programa en un soporte tangible. Para la transferencia de copias digitales (por ejemplo, vía Internet), hablamos de "comunicación pública". Sin embargo, para simplificar la redacción de esta Guía, y salvo mención expresa (sobre todo en el Capítulo 7), utilizaremos los términos "distribución" y "redistribución" para referirnos a las dos formas de diseminar o divulgar una obra. Por otro lado, una "distribución" específica de un programa se refiere a una versión particular del mismo (p.e. Ubuntu 8.2, Linux Kernel 2.6, Firefox 3.0).

- **Dominio público:** situación en la que los derechos de autor sobre el software han caducado por haber terminado el plazo de protección (normalmente, la vida del autor más 70 años), de modo que cualquiera puede explotar el software/programa sin restricciones; o, en Estados Unidos, cuando un titular del software renuncia a ejercer sus derechos y por lo tanto los cede a todos.

- **Freeware / shareware:** software que se distribuye bajo una licencia que permite su uso gratuito y hasta su redistribución, pero no bajo licencia de SFA (en la mayoría de los casos, sin acceso al código fuente ni derecho de transformación).

- **Licencia con copyleft:** licencia que incluye obligaciones de copyleft para la redistribución del programa.





- **Licencia permisiva:** licencia que no impone condiciones de tipo copyleft sobre la redistribución del programa.

- **Obra derivada:** obra que resulta de la transformación de otra, por ejemplo agregando o modificando funcionalidades, o traduciendo su interfaz de usuario.

- **Software de fuentes abiertas (SFA):** software que se distribuye bajo una licencia de fuentes abiertas (open source), en particular una licencia que cumple con las directrices de la Open Source Initiative. En esta guía, sinónimo de software libre.

- **Software libre:** programa que se distribuye bajo una licencia de software que cumple con los criterios anunciados por la Free Software Foundation.

- **Software no libre, privativo o propietario:** software que se distribuye bajo una licencia restrictiva, que no cumple con la definición de software libre o de fuentes abiertas.

Finalmente, el sector de las tecnologías tampoco está exento de siglas:

- **CLUF (o EULA** por sus siglas en inglés): contrato de licencia de usuario final (End-User License Agreement).

- **DRM** (Digital rights Management): gestión de derechos sobre obras digitales.

- **FSF** (Free Software Foundation): Fundación para el Software Libre.

- **OSI** (Open Source Initiative): iniciativa de Software de Fuentes Abiertas.

- **OSD** (Open Source Definition): definición de Software de Fuentes Abiertas.

- **P2P** (peer to peer): redes de pares, para el intercambio o distribución de obras en Internet.

La mayoría de las licencias de software de fuentes abiertas se conocen por sus iniciales (GPL para la licencia GNU General Public License, MPL para la licencia Mozilla Public License, BSD para la licencia Berkeley Software Distribution, etc.). Encontramos una lista de las licencias reconocidas por la OSI en *www.opensource.org/licenses/alphabetical*. En esta Guía utilizaremos estas iniciales salvo cuando no quede claro a qué licencia nos referimos. Asimismo, como algunas licencias tienen varias versiones, será importante identificar el número de versión (GPL versión 3, por ejemplo, o GPLv3).





CAPÍTULO 2

LAS LICENCIAS DE SOFTWARE DE FUENTES ABIERTAS

BSD
Apache
GPL
EUPL
MPL





Ficha resumen

OBJETIVO	Este capítulo presenta brevemente: • La función de las licencias de software. • Las principales licencias de SFA y su clasificación, contrastándolas con licencias privativas. • Las licencias libres para contenidos.
DESTINATARIOS	Qualquier usuario de software bajo licencia de SFA (desarrolladores, integradores o usuarios finales).
RESUMEN	Las licencias de SFA ofrecen una serie de libertades a los licenciatarios: derechos de reproducción, transformación, comunicación pública y distribución del software. Existe una gran variedad de licencias de SFA, que podemos clasificar en: • Permisivas: licencias sin obligaciones sustanciales sobre el uso y redistribución del software. • Con copyleft fuerte: licencias que obligan a usar la misma licencia en el caso de redistribuir el software o programas que lo incluyen. • Con copyleft suave: licencias que tienen copyleft para el núcleo del código pero permiten su la integración de este código en otras aplicaciones bajo licencias distintas (de SFA o privativas).
REFERENCIAS	En esta Guía: • Capítulo 1, sobre la definición de SFA, software libre y copyleft. • Capítulo 3, sobre aspectos prácticos y jurídicos de las licencias de SFA y la selección de una licencia para un proyecto libre. • Capítulo 7, sobre cesiones de derechos de autor. • Capítulo 8, sobre cesiones de derechos de marcas y patentes.
LICENCIAS CITADAS	Las cinco licencias más usadas: BSD, Apache, GPL (v2 y v3), LGPL y MPL así como la EUPL.





2.1. INTRODUCCIÓN: LA LICENCIA DE SOFTWARE

La licencia de software es, según el Derecho español, el contrato por el cual el titular un programa autoriza al licenciatario a utilizarlo, cediéndole los derechos necesarios para este uso.

Licencia de uso

"Utilizar" es un término muy amplio, usado de manera general para referirse a instalar y ejecutar un programa, en algunos casos parametrizarlo, y hasta adaptarlo y redistribuirlo. De hecho, muchas licencias de software estándar se llaman Contrato de Licencia de Usuario Final (CLUF, o EULA por sus siglas en inglés).

En términos jurídicos hablamos de la cesión de derechos de reproducción, transformación, distribución y comunicación pública, que son los actos reservados por la ley al titular del software, de acuerdo con las condiciones establecidas en esta licencia. La cesión podrá cubrir todos o algunos de los derechos de explotación y ser exclusiva o no exclusiva.1 [17]Asimismo puede incluir condiciones relativas al uso de marcas y derechos de patentes.

La licencia de software cumple una doble función:

- asegurar los derechos del usuario (las autorizaciones); y
- reservar y proteger los derechos del titular (los derechos no cedidos y las condiciones que debe cumplir el usuario).

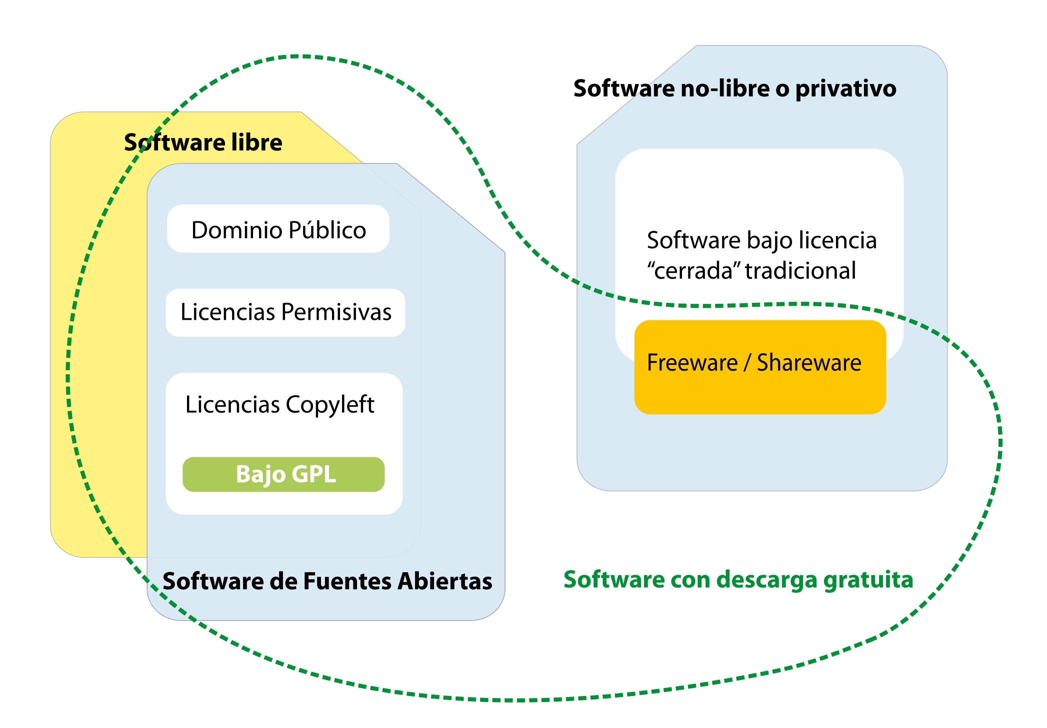
Por lo tanto **la licencia establece determinados derechos y obligaciones entre las partes.** Y es en este punto donde se diferencian las licencias de SFA y las licencias no libres o privativas: mientras que las primeras conceden amplios derechos al usuario (incluidos los de modificar el software y volver a distribuirlo) las segundas suelen limitar o imponer condiciones drásticas.

17 Ver el Capítulo 7 de esta Guía, sobre cesiones de derechos de autor.





Estas diferencias nos permiten clasificar el software según el siguiente diagrama:



Esquema de clasificación del software:
Software[18] de fuentes abiertas - Software privativo

Como vemos en el diagrama, cada conjunto (software libre, no libre) incluye determinados subtipos de licencias (permisivas, copyleft, etc.), que se diferencian entre sí por las condiciones que se establecen en ellas.

2.1.1. Licencias no libres o privativas

Podríamos decir que hay casi tantas licencias no libres como programas de software propietario: software estándar de distribución masiva, como MSWindows® o MacOS®, software empresarial que se parametriza, como SAP®, o software desarrollado a medida para un cliente particular. Las condiciones específicas dependerán del tipo de software, de la posición de las partes que negocian (o no) el contrato, de las jurisdicciones donde se vende el software en cuestión, etc., sin embargo comparten cláusulas similares.

Una licencia de software privativa de usuario final concede, básicamente, el derecho a la instalación del software en el ordenador del usuario y su ejecución, de acuerdo con determinadas condiciones y restricciones.

18 *Adaptado de un diagrama de Chao-Kuei, bajo licencia Creative Commons Attribution-Share Alike v2.0 En http://creativecommons.org/licenses/by-sa/2.0/*





Para ejercer estos derechos, el usuario deberá cumplir una serie de obligaciones. Éstas suelen incluir el pago de derechos de licencia o regalías, la prohibición de la copia, modificación y redistribución[19] del programa, además de otras limitaciones de uso. Pueden limitar la cantidad de procesadores aplicables, de usuarios conectados al software al mismo tiempo, del número de bases de datos a las que se accede o la cantidad de datos procesados. En general, prohíben la descompilación del software y su ingeniería inversa (sujeto a lo que permite la ley en relación a la interoperabilidad).

Dentro de la categoría de licencias privativas se encuentran las de tipo **Freeware** o **Shareware**. Estas licencias suelen permitir el uso gratuito del programa (limitado o no a un período de tiempo, después del cual se deberá pagar un precio). No son licencias de fuentes abiertas porque, aunque permiten copiar y redistribuir el software, no permiten modificarlo ni aseguran el acceso a su código fuente.

2.1.2. Licencias de SFA

En contraste, una licencia de SFA ofrece amplios derechos de uso al licenciatario: para calificarse como "de fuentes abiertas", la licencia debe garantizar al usuario las cuatro libertades del software y/o cumplir las diez directrices de la OSI, enunciadas en la introducción de esta Guía.[20]

Los derechos otorgados bajo cualquier licencia de SFA permiten al licenciatario instalar y ejecutar el programa sin limitaciones (con un uso tanto privado como comercial), reproducirlo y transformarlo, así como distribuir y/o comunicar.

Qué se puede hacer con un software bajo licencia de SFA?

Las licencias de SFA permiten, entre otras cosas:

- Descargar, instalar y ejecutar el software sin limitaciones (en un ordenador, en varios o en una red).
- Hacer una copia de seguridad del software.
- Descargar el código fuente y estudiarlo.
- Analizar las interfaces para hacer un software interoperable.
- Modificar el software para adaptarlo a sus necesidades, recompilar y ejecutarlo.
- Utilizar parte del código para otro software.
- Ampliar el software original con extensiones, plug-ins, nuevas funcionalidades.
- Integrarlo en otro software (SFA) para mejorar sus funcionalidades.
- Redistribuir o comunicar públicamente el software original en una página web, un CD, una o llave USB o, en redes de pares P2P.
- De la misma manera, redistribuir el software modificado y las extensiones (respetando siempre las condiciones de la licencia).

19 Recordemos que usamos este término para referirnos a la distribución en forma tangible o la comunicación pública de los programas en redes digitales.

20 Ver Capítulo 1: Software de fuentes abiertas.





•Crear documentación sobre el software y venderla.

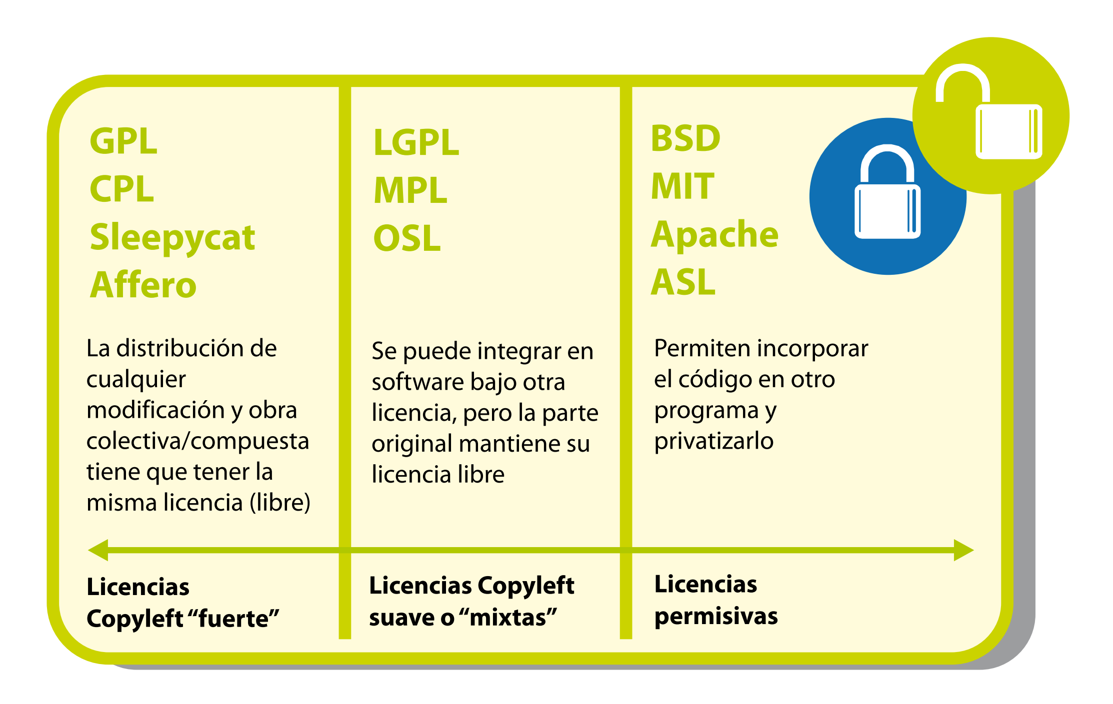
Licencias certificadas y no certificadas.

Es importante resaltar que no todo el software libre o de fuentes abiertas se distribuye bajo una licencia certificada por la OSI o listada por la FSF3[21]. Si bien la mayoría de los proyectos relevantes utilizan licencias conocidas y aceptadas por la comunidad (como la GPL, BSD o MIT) también existe una gran cantidad de software publicado en Internet bajo licencias o términos de uso únicos, establecidos por su titular. Un ejemplo de ello sería la licencia: *"You are free to use this software for any purpose"*.[22]

2.2. TIPOS DE LICENCIA

No todas las licencias de fuentes abiertas son iguales, sino más bien todo lo contrario: las licencias de SFA contienen disposiciones que varían de forma significativa de unas a otras. Hay casi 70 licencias de SFA certificadas por la OSI[23]. Es importante tener en cuenta este hecho siempre que utilicemos SFA y, sobretodo, a la hora de integrar y redistribuir a terceros este tipo de software.

La mayor diferencia entre licencias radica en las condiciones aplicables a la redistribución, , en particular en cuanto al grado de copyleft. En base a esto, las licencias se pueden clasificar de la siguiente manera:



Esquema de licencias de SFA

21 La lista se encuentra en línea en: http://www.gnu.org/licenses/license-list.html
22 Traducción: "Puedes usar este software para cualquier propósito".
23 La lista está disponible en línea en: http://www.opensource.org/licenses/alphabetical.





Aparte del copyleft, las licencias de SFA también se caracterizan por aplicar condiciones adicionales sobre una variedad de temas que sus autores han considerado importantes:

•La prohibición de uso del nombre del titular para promover el software (la licencia Apache Software License, entre otras).
•El alcance de las licencias de patente (MPL, CPL, GPLv3).
•El derecho aplicable y la jurisdicción competente para resolver conflictos (MPL, CPL).
•Variaciones en cuanto a los disclaimers e indemnizaciones (Eclipse Public License).
•La manera de redistribuir el software o sus modificaciones, p.e. en parches (QtPL).
•Los avisos sobre estas modificaciones (MPL, GPL).
•El acceso al código fuente mediante sistemas remotos (OSL, CDDL y Affero GPL).

Son factores que intervendrán a la hora de seleccionar una licencia de distribución de un software.

Otras clasificaciones

Mientras que la FSF clasifica las licencias en libres o no libres y las compatibles con la GPL[24], la OSI, con el objetivo de revisar y reducir la cantidad de licencias en uso y, en consecuencia, facilitar el proceso de selección de las mismas, las clasifica en[25]:
•Populares
•Específicas
•Otras
•Redundantes (candidatas a eliminación)
•No reutilizables (asociadas con sus autores o un producto)
•Sustituidas
•Sin clasificación todavía (donde encontramos la GPLv3 y LGPLv3 conviviendo, paradójicamente, con las licencias de fuentes abiertas de Microsoft y la EUPL).

2.2.1. Licencias permisivas o académicas

Las licencias de SFA permisivas se llaman así porque no imponen ninguna condición particular a la redistribución del software, salvo la de mantener los avisos legales y las limitaciones de garantías y responsabilidades. Por ello es posible integrar y redistribuir software bajo licencias permisivas en aplicaciones bajo cualquier otro tipo de licencia, ya sean de SFA o de software propietario.

Este tipo de licencia es el resultado del deseo de sus autores de compartir el software con cualquier fin, sin imponer obligaciones que puedan restringir los usos tanto personales como comerciales, libres o privativos.

Las más conocidas son las licencias **Berkeley Software Distribution (BSD) y Apache Software License (ASL)**, comentadas más abajo en el Apartado 3, así como la licencia **MIT**.



24 En http://www.fsf.org/licensing/licenses/
25 En http://opensource.org/licenses/category





Origen de las licencias permisivas

Originalmente, la mayoría del software distribuido bajo estas licencias era fruto de las investigaciones y los trabajos universitarios financiados por el gobierno (de los EEUU, en particular). Por lo tanto, debía ser de acceso libre para todos y solamente debían protegerse los derechos morales de los autores (por la simple obligación de mantener los avisos de titularidad: el copyright notice) y limitar eventuales responsabilidades (con los disclaimers). Es por esta razón que, a veces, se denomina a estas licencias "académicas" y llevan el nombre de una universidad (Univ. of California, Berkeley o MIT), o más genéricamente la Education Community License[26]. La Apache Software License, que comentamos más adelante, surge del proyecto Apache del National Center for Supercomputing Applications, de la Universidad de Illinois.

La lista de licencias de SFA permisivas es extensa, por lo que no las recogemos todas aquí: Zope Public License version 2.0, Open LDAP License 2.7, Artistic Licence, Perl License, Academic Free License, PHP 3.0, Qt Public License 1.0, etc.

Versiones más recientes y más sofisticadas de estas licencias, como la Apache Software License 2.0, agregan cláusulas sobre patentes o el uso de marcas, que no afectan a este principio general de libertad a la hora de redistribuir el software.

2.2.2. Licencias con copyleft fuerte

Las licencias de SFA con copyleft fuerte (a las que a veces algunas personas se refieren negativamente como víricas), como la GPL, son las que exigen el uso de la misma licencia para cualquier redistribución del programa y de las modificaciones que se realicen del mismo, así como a programas que lo utilizan o incorporan (en forma de librerías, etc.). Este último punto es lo que las diferencia del copyleft suave. Asimismo, el licenciatario debe dar al usuario una copia del código fuente u ofrecerle un medio para obtenerlo (un repositorio en línea, por ejemplo).

Su objetivo básico es asegurar que cualquier usuario (directo o indirecto) del software siempre tenga acceso al código fuente, bajo los términos de esta misma licencia. Como consecuencia, se impide la distribución del software con *copyleft* en aplicaciones privativas (lo que se ha llamado *"privatizarlo"* o *"cerrarlo"*). Ello no significa que no se puedan crear y vender aplicaciones comerciales con software copyleft. Pero sí será una violación de la licencia redistribuir este software bajo otra licencia, distinta, que requiera, por ejemplo, el pago de regalías y/o que impida la modificación del software por parte de su destinatario.

Es importante destacar que sólo es obligatorio divulgar el código fuente de un programa bajo licencia copyleft fuerte (y/o sus modificaciones) en el momento de su redistribución.

26 *Texto original en http://opensource.org/licenses/ecl2.php*





La **GNU General Public License (GPL)**[27] es la licencia de SFA con copyleft fuerte más conocida. Redactada y promovida por la FSF, es la licencia usada por aproximadamente el 75% de los proyectos disponibles en Sourceforge. Hoy tiene dos versiones activas: la Versión 2 (de 1991) y la nueva Versión 3 (de 2007). Otras licencias con copyleft fuerte son la licencia Affero GPL, o la licencia Sleepycat.

Copyleft y servicios remotos o ASP

Como ya hemos comentado, el efecto copyleft se activa con la redistribución del software. Pero con los servicios de software remoto no hay ninguna distribución del código, sino de sus funcionalidades o usos. Por esta razón, la persona que ofrece estos servicios basados en software GPL (como Google y muchos otros) no están sujetos a la obligación de proporcionar el código (binario o fuente) a los usuarios finales. Para algunos, esto es una manera de evitar las obligaciones de copyleft y un fallo de la licencia GPL.

La licencia Affero GPL (o AGPL) es una variante de la GPL que corrige lo que se ha considerado como fallo, incluyendo la obligación de proporcionar también a los usuarios remotos del software una copia del programa y su código fuente (integrando, por ejemplo, un botón o función de descarga) bajo esta licencia.

2.2.3. Licencias mixtas o con copyleft suave

Las licencias de SFA mixtas incluyen cláusulas de copyleft sólo para el código original, sin que esto afecte a otros programas que lo integren (con o sin modificaciones) o lo utilicen, por ejemplo como librería. Esta dilución del copyleft se efectúa, básicamente, de dos maneras:

•Permitiendo el uso del software (como librería) por programas que se distribuyan bajo una licencia diferente (la ***Lesser GPL*** o ***LGPL***), o

•permitiendo su incorporación en una obra más amplia (u "obra mayor") cuya licencia, igualmente, puede ser distinta (las licencias ***Mozilla Public License (MPL) y CDDL***[28], entre otras).

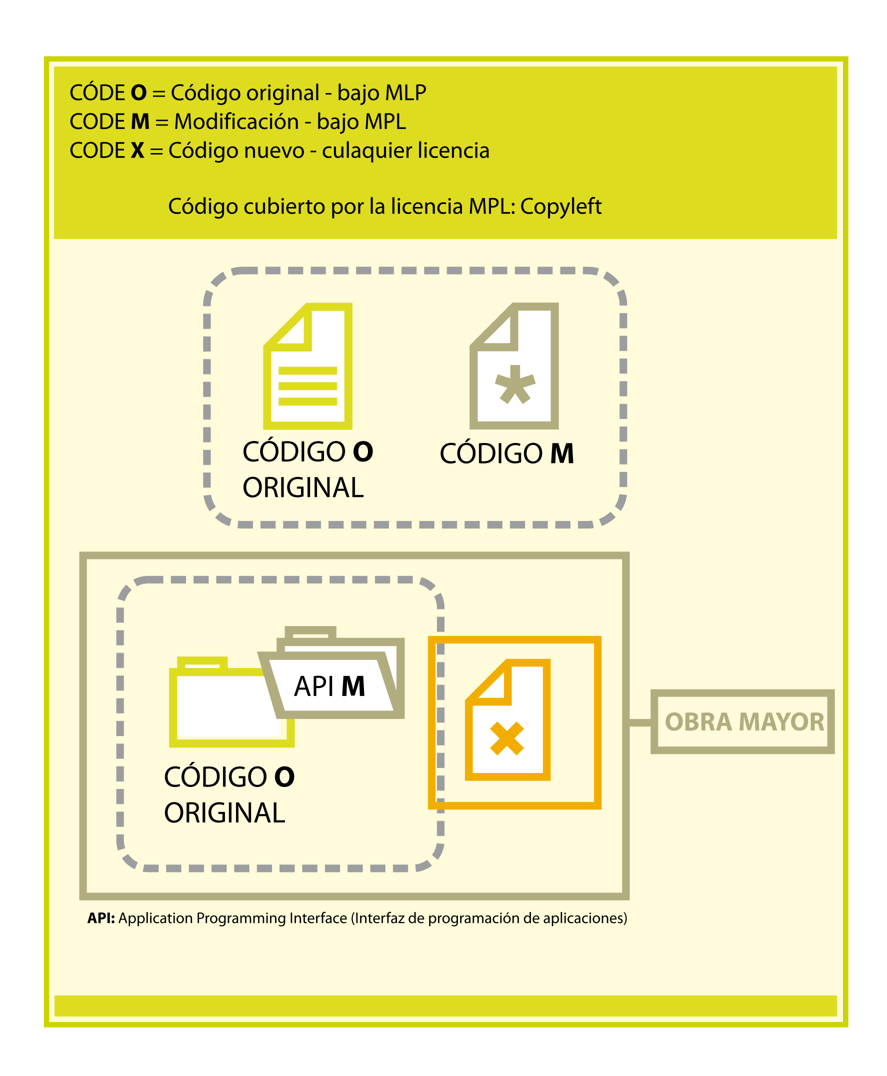
En ambos casos está permitido agregar y/o vincular un nuevo programa al código original, y distribuir el conjunto bajo una nueva licencia (privativa o libre). Sin embargo, la parte original y sus modificaciones tendrán que distribuirse bajo su licencia inicial y, normalmente, junto con el código fuente. Este mecanismo puede ilustrarse de la siguiente manera (respecto de la MPL):

27 *Incluimos una ficha sobre la GPL en Apartado 3 de este capítulo.*
28 *Common Development and Distribution License, en http://opensource.org/licenses/cddl1.php*





Esquema de la licencia Mozilla Public License



Estas licencias se usan mucho para SFA de tipo empresarial porque permite que otros profesionales del sector (integradores, productores de software de tipo ISV) agreguen extensiones o módulos para sus clientes finales bajo cualquier licencia.[29]

Otras licencias con *copyleft suave*

La MPL es modelo de otras licencias de SFA como la CDDL o la reciente CPAL, cada una con sus características particulares:[30]

La CDDL es una adaptación de la MPL realizada por Sun Microsystems Inc. que amplia la definición de "distribución" (y por lo tanto, el alcance del copyleft), obligando a poner el código fuente a disposición de los usuarios del software en modo de acceso remoto o ASP.

La CPAL 1.0 es una variante de la MPL que permite al licenciante obligar al usuario a mantener un aviso de origen ("powered by" o similar) en las interfaces de usuario del software.

29 Por ejemplo, proyectos como Pentaho, SugarCRM (hasta versión1.2), Alfresco (hasta versión 2), Openbravo ERP, Socialtext, etc.
30 Las licencias están en: CDDL. http://opensource.org/licenses/cddl1.php, CPAL: http://opensource.org/licenses/cpal_1.0 y OSL 3.0: http://opensource.org/licenses/osl-3.0.php



La OSL 3.0 es una licencia de SFA que se acerca al marco legal europeo en cuanto al derecho de la propiedad intelectual y garantías y responsabilidades se refiere. En este caso, el copyleft sólo afecta a las "obras derivadas según la definición del derecho de la propiedad intelectual que se aplique en cada caso". Esto permite, generalmente, enlazar software bajo OSL (como librerías o mediante vínculos dinámicos) con otros programas sin que la licencia afecte a su distribución. Por lo que se refiere a servicios provistos por el software (en modo acceso remoto o ASP) el alcance del copyleft es similar al de la licencia CDDL.

2.3. SEIS LICENCIAS DE SOFTWARE DE FUENTES ABIERTAS

Esta sección contiene un breve resumen de las principales características de las licencias de SFA más usadas (comentadas aquí a modo de fichas):[31]

- La Berkeley Software Distribution (**BSD**) modificada: licencia permisiva.
- La Apache Software Licence 2.0 (**ASL 2.0**): licencia permisiva.
- La GNU General Public License (**GPL v2 y v3**): licencia con copyleft fuerte.
- La GNU Library or Lesser General Public License (**LGPL v2 y v3**): licencia con copyleft suave, o mixta.
- La Mozilla Public License 1.1 (**MPL 1.1**): licencia con copyleft suave, o mixta.
- La licencia European Unión Public License (**EUPL 1.1**): licencia con copyleft suave, o mixta, explícitamente compatible con ciertas licencias copyleft más usadas.

2.3.1. La licencia *Berkeley Software Distribution* (BSD modificada)

La ***new-BSD*** (o "***modified BSD***") es quizá la más simple y permisiva de las licencias libres, junto con su hermana, la MIT[32]. Surge de las distribuciones de versiones de Unix de la Universidad de California, Berkeley, en los años 80, cuando se inicia el movimiento del FOSS. Esta licencia permite el uso del software sin límites y su redistribución en forma de código fuente y binario. Por lo tanto, se puede modificar e integrar en otros programas sin restricciones, incluso propietarios.

31 Esta sección incluye un resumen general, a modo de "referencia", de algunas de las licencias más conocidas. Para su interpretación o aplicación a cada caso real será necesario leerlas con detenimiento y contar con asesoramiento legal si fuera necesario.

32 Textos originales en Internet: BSD: http://opensource.org/licenses/bsd-license.php; MIT en http://opensource.org/licenses/mit-license.html





Licencia BSD-modificada (o BSD nueva)

Derechos otorgados

•La licencia BSD otorga los derechos de "redistribución y uso, con o sin modificación, en forma de código fuente o binario". Se entiende que permite la reproducción, transformación, distribución y comunicación pública del software en ambas formas.

Obligaciones

•Se debe mantener el aviso de titularidad y las exclusiones de garantías y responsabilidades en el código fuente. Si se redistribuye el software en forma de código objeto, el aviso de titularidad y las exclusiones deben estar en la documentación.

•Se prohíbe el uso del nombre del titular o contribuidores para la promoción de un software que incluya el programa o se base en él.

Otros

•Se excluyen garantías y responsabilidades.

•No se hace referencia a patentes o marcas, ni al derecho aplicable o jurisdicción competente.

Comentarios

•Compatibilidad con otras licencias: la permisividad de la licencia BSD hace que sea compatible con casi cualquier otro SFA. Permite su integración en software bajo cualquier licencia, hasta licencias privativas.

2.3.2. La Apache Software License 2.0 (ASL 2.0)

Licencia permisiva y moderna (de enero del 2004), la ASL 2.0 es una nueva versión de las Apache 1.0 y 1.1, licencias permisivas y simples basadas en la BSD.

Se usa para distribuir el software de la Fundación Apache, entre muchos otros programas disponibles en Internet.





Apache Software License 2.0

Derechos otorgados

•La licencia ASL otorga los derechos de reproducción, modificación, distribución y comunicación pública (display/perform) además del de sublicenciar en forma de código fuente o binario.

•Asimismo, la ASL otorga una licencia de patente, que se revoca en caso de que el licenciatario inicie demandas basadas en patentes contra el titular.

Obligaciones

•Se debe mantener una copia de la licencia, el aviso de titularidad y las exclusiones de garantías y responsabilidades en el código fuente. Si se redistribuye el software en forma de código objeto, esta información debe estar en la documentación.

•A la hora de redistribuir el software, se debe indicar cualquier modificación realizada y mantener el fichero "notice.txt" (un fichero con comentarios jurídicos) con el código o la documentación.

•Hay que conservar cualquier aviso referente a las marcas, aunque prohíbe el uso del nombre y las marcas del licenciante para promocionar cualquier software derivado del programa.

Otros

•Las garantías y responsabilidades quedan excluidas en la medida permitida por ley aplicable.

•No se hace referencia al derecho aplicable o jurisdicción competente.

Comentarios

•Compatibilidad con otras licencias: la ASL es compatible con la mayoría del SFA. La FSF considera que la ASL 2.0 es incompatible con la licencia GPL 2.0 (ya que la licencia de patente es una restricción adicional prohibida por ésta), pero compatible con la GPLv3.

2.3.3. La GNU General Public License (GPL)

Como cualquier licencia de SFA, la GPLv2 permite la reproducción, transformación, comunicación pública y distribución del software, en forma de código fuente o binario. Es la licencia copyleft por antonomasia. Es importante recordar que la obligación de ofrecer acceso al código fuente del software y/o sus modificaciones surge en el momento de la redistribución. Tiene dos versiones activas que comentamos en las siguientes tablas.





La GPLv2 o 2.0

Derechos otorgados

•La GPL 2.0 otorga los derechos de reproducción, modificación y distribución del programa en forma de código fuente o binario.

Obligaciones

•Se debe mantener los avisos de titularidad e indicar si se ha modificado el programa.

•Se debe usar la misma licencia en caso de redistribuir el programa cualquier modificación del mismo o integrarlo en otro software (efecto copyleft fuerte).

•Si la distribución es en forma binaria, el código fuente debe distribuirse junto con el binario o estar disponible para cualquier persona durante tres años.

•No se permite agregar ninguna restricción adicional sobre la distribución del programa.

Otros

•El código fuente se define como la "forma preferida de la obra para hacerle modificaciones" e incluye los scripts para su compilación y ejecución, así como cualquier API.

•No se hace referencia a patentes o marcas, ni al derecho aplicable o jurisdicción competente.

•Las garantías y responsabilidades quedan excluidas en la medida permitida por ley aplicable.

Comentarios

•La FSF, autora de la licencia, argumenta que ésta no permite enlazar mediante vínculos dinámicos un programa cualquiera a otro licenciado bajo la GPL sin aplicar esta misma licencia al conjunto del software "como un todo" (aunque ésta no es una opinión compartida en toda la comunidad libre). Sin embargo, el efecto copyleft de la GPL no se extiende a programas agregados y/o distribuidos en un mismo soporte (por ejemplo, en un CD/DVD).

•Compatibilidad con otras licencias: el programa debe redistribuirse siempre bajo la misma licencia, por lo que es solamente compatible con las más permisivas (BSD, MIT, etc.). Hay una lista en www.gnu.org.

•Hay variantes de la GPL con "excepciones" que aumentan su compatibilidad con otras licencias, como la "Excepción Classpath" del proyecto Classpath (que permite la vinculación dinámica con otros programas bajo cualquier licencia), o la "Excepción FOSS" de MySQL (que permite lo mismo, siempre que el otro programa se distribuya bajo licencia de SFA).

•Los programas que utilizan esta licencia suelen indicar que se trata de la "GPLv2 o cualquier versión posterior". Esto permite al usuario recibir el programa bajo la licencia GPLv3 si así lo prefiere.





La GPLv3

La licencia GPLv3 es el resultado de un proceso comunitario de revisión de la GPLv2 que duró más de dos años, a lo largo de los cuales se celebraron cinco conferencias internacionales (la tercera en Europa, Barcelona[33]) para comentar los distintos borradores que se fueron generando. Es una licencia bastante más completa, pero también más compleja (por su alcance: derechos de autor, patentes, marcas, etc.).

Modernización de la GPLv2

El proceso de modernización tenía los siguientes objetivos principales:

•**Internacionalizar:** la GPLv3 trata de extender la validez internacional de la licencia. Usa un vocabulario internacional no sujeto a interpretaciones jurídicas preexistentes, y limita el alcance de los disclaimers para que se adapten a lo "permitido por derecho aplicable".

•**Modernizar:** la GPLv3 es más actual en cuanto a las referencias tecnológicas. Contempla, por ejemplo, la distribución del código desde servidores de Internet y redes P2P, o la integración de componentes mediante enlaces dinámicos.

•**Flexibilizar (parcialmente):** la nueva versión es compatible con más licencias (la ASL 2.0 en particular). Permite al usuario:

- Agregar o eliminar derechos y algunas restricciones adicionales.
- Subsanar posibles incumplimientos en un período de 30 días.
- Externalizar los servicios informáticos sin activar el copyleft.

•**Responder a los sistemas *DRM*[34]:** trata de evitar cualquier impacto que puedan tener estos sistemas e impide indirectamente su uso con software bajo licencia GPL (en particular la Tivo-ización[35]).

•**Responder a las patentes de software:** la GPLv3 concede una licencia de patente sobre el software, como pasa con otras licencias modernas.

33 Transcripción en http://www.fsfeurope.org/projects/gplv3/europe-gplv3-conference.es.html
34 Digital Rights ("Restrictions", según la FSF) Management System: medida tecnológica de protección de los derechos.
35 Un sistema bajo el cual un distribuidor cumple con la licencia y permite la modificación del software, pero luego estas modificaciones no funcionarán con el resto del programa por no ser firmadas correctamente con las claves necesarias, mermando por tanto la libertad del usuario.





GNU GPLv3

Derechos otorgados

La GPLv3 concede el derecho de "propagar" el programa, con o sin modificaciones, y de obras basadas en mismo. Se define propagación como la realización de cualquier acto que requiera el consentimiento del titular (en España, los derechos de reproducción, transformación, comunicación pública y distribución).

Otorga una licencia llimitada de patente.

Permite enlazar el programa con software bajo licencia Affero GPL.

Con el objetivo de flexibilizar y augmentar el grado de compatibilidad con otras licencias, permite integrar software con restricciones adicionales al software bajo la GPLv3 (sólo en relación a determinados aspectos: marcas, indemnizaciones, garantías, etc.)..

También es posible agregar permisos adicionales (como en la LGPL v3) que pueden eliminarse posteriormente.

Obligaciones

Se debe mantener los avisos de titularidad, e indicar si se ha modificado.

Se debe usar la misma licencia en caso de redistribuir el programa cualquier modificación del mismo o integrarlo en otro software (efecto copyleft fuerte).

Si la distribución es en forma binaria, el código fuente debe distribuirse con el binario o estar disponible para el licenciatario durante tres años.

No se permite agregar ninguna restricción adicional sobre la distribución del programa.

Otros

No se hace referencia a derecho aplicable o jurisdicción.

Las garantías y responsabilidades quedan excluidas en la medida permitida por ley aplicable.

Comentarios

El código fuente se define de forma más concreta y extensa que en la GPLv2, pero básicamente recoge la idea de la "forma preferida de la obra para hacerle modificaciones" e incluye los scripts para su compilación y ejecución, así como cualquier API. En caso de productos de consumo, también se deben incluir los códigos o claves necesarias para hacer funcionar el software.

Se especifican nuevas formas de hacer accesible el código fuente, incluidos por un sitio web o por las redes P2P.

Se explicita que el copyleft afecta a aquellos programas vinculados dinámicamente a software bajo la GPLv3. Sin embargo, como en la GPLv2, el copyleft de la GPLv3 no se extiende a programas agregados y/o distribuidos en un mismo soporte (por ejemplo, en





un CD/DVD). Como excepción, el copyleft no se aplica a distribuciones del programa y modificaciones entre un consultor-integrador y su cliente.

Compatibilidad con otras licencias: hay un esquema complejo de compatibilidad con software bajo la GPLv2 y LGPL. Las "restricciones permitidas" hacen que sea compatible con más licencias (Apache 2.0 en particular).

Aunque permite incluir el software bajo GPLv3 en sistemas de control y protección de derechos de autor (DRMS), no permite al titular prohibir la elusión de las medidas tecnológicas de protección por los licenciatarios.

La licencia se revoca en caso de que el licenciatario inicie demandas basadas en derechos de patente, y contiene condiciones adicionales en caso de pactar con terceros una protección particular frente a las patentes.

Debido a su reciente publicación, el impacto que tendrá en el mundo del software libre es todavía incierto. La GPL sigue un proceso de adopción progresiva entre los miembros de la comunidad. El sistema de licenciar el software bajo "la GPL versión 2 o versiones posteriores" permite a los licenciatarios elegir versiones futuras de la licencia y por tanto facilitará, sin duda, la adopción de la GPLv3 por parte de usuarios y redistribuidores.

2.3.4. La Lesser GPL (LGPL)

La LGPL se creó específicamente para permitir que componentes bajo esta licencia se enlazaran con programas bajo otra licencia distinta, de SFA o no libre. Por lo tanto, la LGPL es una licencia cómoda y atractiva para los desarrolladores de aplicaciones bajo licencias incompatibles con la GPL (inclusas las privativas) que quieren enlazar sus programas con otros componentes bajo licencias libres (a modo de librerías), pero que temen el efecto copyleft de la GPL.

En términos generales, la licencia LGPL incluye las mismas libertades y restricciones que la GPL al software originario bajo esta licencia: la explotación de cualquier modificación del software estará cubierta por el copyleft. Pero, por otro lado, permite aplicar condiciones legales diferentes (incluso condiciones restrictivas o no-libres) a la distribución de "programas que usan la librería".

La LGPLv2[36] tiene una redacción muy compleja en cuanto a enlaces entre programas.

La LGPLv3[37] es una variante explícita de la GPLv3. Es decir, es la licencia GPLv3 pero con permisos adicionales que aceptan que un tercer programa, bajo cualquier licencia, pueda usar el software bajo esta licencia, además de permitir la distribución de todo el conjunto bajo una licencia que no sea la LGPL.

36 En Internet en: http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html
37 En Internet en: http://www.gnu.org/copyleft/lgp.html





Combinaciones de programas con la LGPLv3

Según la LGPLv3, la combinación de programas puede ser:

•Por usar los datos de cabecera de la librería (mediante enlace dinámico).
•Por combinar los programas juntos en una sola aplicación.
•Por poner juntas las funciones de la librería original y otras funciones nuevas en una librería combinada.

2.3.5. La Mozilla Public License (MPL)

La licencia MPL[38] fue creada en el momento de liberar el navegador Netscape, en 1998, como resultado de un proceso que acabó en la redacción de la Netscape Public License (para la distribución del navegador en cuestión) y la Mozilla Public License (para futuros programas basados en el código original de la empresa).

Hoy en día, la MPL 1.1 no se utiliza únicamente para los productos del proyecto Mozilla (Mozilla, Firefox, Thunderbird, etc.), sino también para muchas otras aplicaciones libres, en particular SFA empresarial patrocinado por empresas privadas.

Mozilla Public License 1.1

Derechos otorgados

•La licencia MPL otorga los derechos de reproducción, modificación y distribución del programa, en forma de código fuente o binario.

•Asimismo, otorga una licencia de patente, que se revoca en caso de que el licenciatario inicie demandas basadas en esta figura jurídica en relación a cualquier software del titular.

Obligaciones

•Se debe mantener una copia de la licencia, el aviso de titularidad y las exclusiones de garantías y responsabilidades en el código fuente.

•A la hora de redistribuir el software, se debe indicar cualquier modificación realizada y mantener el fichero "legal.txt" (un fichero con comentarios jurídicos) con el código o la documentación.

•Efecto copyleft suave: a la hora de redistribuir el software o cualquier modificación (de los ficheros originarios), se debe usar la misma licencia y proporcionar al destinatario el código fuente. Sin embargo, se puede integrar el programa en otro software distribuido bajo cualquier licencia.

38 En Internet en: http://www.mozilla.org/MPL/MPL-1.1.html





Otros

La licencia se somete el derecho californiano y los tribunales de Santa Clara, California.

•Las garantías y responsabilidades quedan excluidas en la medida permitida por ley aplicable.

•Permite al titular especificar el uso de otras licencias para partes o todo el software ("dual licensing").

Comentarios

•Compatibilidad con otras licencias: la MPL es compatible con la mayoría de licencias permisivas y mixtas, pero no lo es con cualquier licencia copyleft, en particular con la GPLv2 o v3.

2.3.6. La European Union Public License (EUPL)

La **EUPL**[39] o European Union Public License (Licencia Pública de la Unión Europea) es una nueva licencia (de enero del 2007) redactada expresamente para la liberación de software de la administración pública europea y de los países miembros de la UE. El alcance del copyleft es similar al de la OSL 3.0.

European Union Public License 1.1

Derechos otorgados

•La licencia EUPL otorga los derechos de reproducción, transformación, comunicación pública, distribución, sub-licencia, alquiler y préstamo, sobre el programa y sus obras derivadas, en forma de código fuente o binario.

•Asimismo, la EUPL otorga una breve licencia de patente para el uso del programa.

Obligaciones

•Se debe mantener los avisos de titularidad en el código fuente e indicar las modificaciones.

•A la hora de redistribuir el software, se debe indicar cualquier modificación realizada y mantener el fichero "legal.txt" (un fichero con comentarios jurídicos) con el código o la documentación.

•Efecto copyleft suave: a la hora de redistribuir el software o cualquier modificación (de los ficheros originarios) se debe usar la misma licencia y proporcionar al destinatario el código fuente o, en su defecto, indicar dónde se puede conseguir fácil y gratuitamente.

39 Versiones en http://ec.europa.eu/idabc/en/document/7330





Otros

•La licencia se somete el derecho del proveedor del software (o Bélgica, para software de la Comisión o Unión Europea).

•El licenciante ofrece una garantía de titularidad. El resto de garantías y responsabilidades quedan excluidas en la medida permitida por ley aplicable.

•La distribución por medios telemáticos debe asegurar la aceptación de la licencia comediante un "clic" y el proveedor está obligado a publicar la información legal obligada por ley en el sitio web.

Comentarios

•Compatibilidad con otras licencias: la EUPL es compatible con la mayoría de licencias permisivas y mixtas (copyleft suave). Además, incluye una cláusula de compatibilidad expresa: en el caso de combinar software bajo la EUPL con software bajo alguna de las licencias mencionadas en el anexo (GPLv2, CPLv1, EPLv1, OSL v3, CeCILLv1), el resultado se podrá distribuir bajo la licencia del otro software.

•Efecto copyleft suave: se considera que la licencia permite integrar un programa bajo esta licencia en otro software distribuido bajo cualquier licencia.

La Comisión Europea ha traducido y publicado esta licencia en las lenguas oficiales de la UE (es la única licencia FOSS con traducciones oficiales). En febrero del 2009 la OSI certificó la licencia EUPL 1.1. como **"***open source***".**

2.4. COMENTARIOS FINALES

El derecho de propiedad intelectual e industrial ha venido protegiendo las obras e invenciones con una determinada filosofía restrictiva[40], reforzada por las condiciones establecidas en las licencias privativas.

En esta sección, hemos mostrado cómo el movimiento de software libre y de fuentes abiertas ha creado una serie de mecanismos jurídicos (las licencias de SFA) para conseguir efectos liberadores en este marco restrictivo, con el fin de ofrecer a todo el mundo lo que la ley reserva, originalmente y de manera exclusiva, a los titulares del software.

Estos mecanismos, aunque puedan parecer incompatibles con el actual marco jurídico, están igualmente amparados por la ley, buscando un nuevo equilibrio que favorezca la creatividad y la difusión de conocimientos.

40 *Tal y como lo comentamos con más detalles en los capítulos 7 y 8.*





CAPÍTULO 3:

LAS LICENCIAS DE SFA EN LA PRÁCTICA





Ficha resumen

OBJETIVO

Comentar algunos aspectos jurídicos que afectan al uso de las licencias SFA en la práctica.

DESTINATARIOS

Usuarios de software bajo licencia de SFA, y desarrolladores e integradores de SFA.

RESUMEN

Este capítulo se centra en tres temas particularmente relevantes:

la compatibilidad jurídica entre licencias a la hora de integrar y redistribuir SFA.

la selección de licencia para nuevos desarrollos, y

la posibilidad de distribuir un software bajo varias licencias (licencias duales o dual licensing).

También trataremos cuestiones más generales de carácter jurídico como son la naturaleza de las licencias de SFA, la validez de sus condiciones y de las exclusiones de garantías y responsabilidades, y cómo hay que proceder para hacer cumplir los términos de las mismas.

REFERENCIAS

En esta Guía:
Capítulo 2, sobre licencias de SFA.
Capítulo 4, sobre los derechos de los usuarios.
Capítulos 6, sobre la liberación de software.

En Internet:
Licencias duales:
http://en.wikipedia.org/wiki/Dual-licensing
http://www.oss-watch.ac.uk/resources/duallicence.xml

Selección de licencia:
http://www.rosenlaw.com/Rosen_Ch10.pdf





3.1 INTRODUCCIÓN

Esta sección tiene el objetivo de dar respuesta a algunas de las dudas y cuestiones jurídicas que surgen habitualmente a la hora de usar o aplicar licencias de SFA.[41]
A la hora de trabajar con SFA, a menudo surgen tres preguntas prácticas:

1. ¿Cómo se determina la compatibilidad entre licencias de SFA? (Licencias incompatibles: cuando dos licencias no permiten que sus respectivos programas puedan distribuirse integrados en un mismo software)

2. ¿Cuál es la licencia más adecuada para mi proyecto de SFA? Ello dependerá, en gran parte, de los objetivos y motivaciones del proyecto, así como de los componentes del mismo.

3. ¿Cómo puedo distribuir un SFA bajo dos o más licencias? Para ello es necesario conocer y entender el régimen de licencia dual.

Finalmente trataremos otras cuestiones de naturaleza más jurídica, pero no menos importantes, relacionadas con las licencias de SFA.

3.2. TRABAJAR CON LICENCIAS DE SFA

3.2.1. La compatibilidad entre licencias de SFA

Las licencias de SFA imponen distintas condiciones a la hora de explotar el software, sobre todo por lo que se refiere a su redistribución[42]. Estas diferencias pueden dar lugar a una incompatibilidad jurídica entre ellas. Esto no supone ningún problema a la hora de integrar y usar componentes de manera interna (sin distribución a terceros), pero impedirá, en la mayoría de los casos, la posterior redistribución de componentes bajo licencias incompatibles integrados en un mismo producto.

En el contexto del SFA, entendemos que dos o más licencias son compatibles cuando se puede integrar el código de un programa en otro bajo una licencia distinta sin que la redistribución de la obra resultante infrinja la licencia del primero. Por ejemplo, se puede incorporar software bajo una licencia permisiva como la BSD en un programa bajo la GPL, porque la redistribución de los dos componentes juntos no infringirá la licencia BSD.

En caso contrario, las licencias serán incompatibles.

41 En cuanto a las preguntas más generales respecto al SFA como: ¿Quién me dará soporte? ¿Qué continuidad tiene el desarrollo del software? O ¿Puedo consultar referencias a otros clientes?, dirigimos al lector a la Guía Breve del Software Libre, publicada en esta misma colección por el CENATIC.

42 Ver nuestros comentarios sobre las licencias de SFA en el Capítulo 2.





Aplicaciones SFA complejas

Los paquetes de SFA de tipo empresarial son el típico ejemplo de aplicación que incorpora componentes bajo distintas licencias (con funcionalidades implementadas por librerías y otros subcomponentes): Zimbra Collaboration Suite, Alfresco DMS, Openbravo ERP, Pentaho Business Intelligence, etc[43].

A continuación exponemos algunas reglas que serán de utilidad a la hora de identificar una posible incompatibilidad entre licencias:

•Si todos los componentes de una aplicación se reciben y se distribuyen bajo la **misma licencia**, entonces no habrá ninguna incompatibilidad.

•La integración de componentes bajo licencias permisivas en otros programas (bajo cualquier otra licencia) no debería ser problemático, ya que generalmente permiten cualquier tipo de régimen de redistribución (incluso su integración en software propietario).

•El conflicto surge a la hora de distribuir componentes bajo dos o más **licencias *copyleft***. Cada uno de ellos exigirá el uso de la propia licencia para redistribuir el software. Entonces ¿qué licencia debe usarse para la distribución del resultado de la integración de los componentes?

Efectos de las condiciones de licencia

Una condición esencial de la GPLv2 para la redistribución del código original u obras derivadas es la de "no agregar más restricciones" que las incluidas en la misma licencia. Por lo tanto, cualquier licencia que imponga nuevas restricciones (por ejemplo, condiciones en cuanto a derecho aplicable, a la prohibición de uso de marcas, a indemnizaciones, etc.) será incompatible con la GPLv2. Notemos que la nueva versión de la GPL, la versión 3, incorpora diferentes mecanismos legales para mejorar la compatibilidad entre licencias.[44]

Aún así, la compatibilidad o no entre licencias dependerá, siempre, de la manera de integrar los distintos componentes y de si esta integración constituye, desde la perspectiva jurídica, una transformación o no de los mismos. Como el diseño y la arquitectura de cada programa es diferente, para determinar las compatibilidades entre licencias y evitar posibles infracciones será frecuentemente necesario estudiar cada caso en particular, siendo recomendable acudir a profesionales especializados en caso de duda.

43 En www.zimbra.com, www.alfresco.org, www.openbravo.com, www.pentaho.com respectivamente.

44 Las FAQs sobre la la GPLv3 muestran una tabla compleja sobre la compatibilidad entre las diferentes versiones de la GPL y LGPL, en http://www.gnu.org/licenses/gpl-faq.html#AllCompatibility, así como un diagrama interesante en http://www.gnu.org/licenses/quick-guide-gplv3.html para la incorporación de software bajo licencia permisiva.





Ejemplos de compatibilidad / incompatibilidad:

1. Licencias permisivas: mezclar o integrar un componente bajo licencia permisiva (como la BSD modificada o la MIT) en un programa bajo licencia con copyleft (GPL, MPL, CPL, etc.) no suele suponer ningún problema. La licencia con copyleft cumplirá con las pocas obligaciones impuestas por la licencia permisiva a la hora de redistribuir el resultado: mantener los avisos de autoría y los disclaimers (limitaciones de garantías y responsabilidades).[45]

2. Copyleft fuerte: licencias como la GPLv2 prohíben restricciones adicionales, que no se aplican solamente al ejercicio de los derechos de autor. Esto hace que no pueda integrarse software bajo la GPL con software bajo licencias que imponen condiciones específicas con respecto al derecho aplicable o la jurisdicción competente (la CPL impone Nueva York, la MPL California), sobre la manera de distribuir obras derivadas (la QPL), el alcance de las de licencias de patentes (Apache 2.0, MPL, CDDL), obligaciones en cuanto a la documentación (XFree86) o publicidad (BSD original, Apache 1.0), obligaciones adicionales para la provisión del código fuente (Affero), y un largo etc.

3. Copyleft suave: un software que utiliza librerías bajo la LGPL puede ser distribuido bajo otras licencias, como la ASL o MPL. Asimismo, la LGPL tiene un pacto de compatibilidad expresa con la GPL (en este caso, la licencia de la librería se convierte a GPL).

Recordemos que el problema de incompatibilidad entre licencias **solamente surge cuando se redistribuye el resultado**, pues ninguna licencia de fuentes abiertas impone condiciones sobre el uso y modificación por parte del usuario final o dentro de una organización.

3.2.2 La selección de una licencia

La selección de una licencia de distribución es una decisión que se debe tomar, cuanto antes mejor, en el desarrollo de cualquier proyecto de SFA. Esta decisión determinará la compatibilidad con otras licencias (y, por lo tanto, la posibilidad de integrar otros componentes en el producto) así como cuestiones más estratégicas como los medios o potenciales vías de comercialización del software y de productos o servicios basados en el mismo.

Esto no será fácil si tenemos en cuenta las casi 70 licencias de SFA reconocidas por la OSI. Sin embargo, una reflexión sobre cuáles son los objetivos del proyecto y otros factores relevantes que enumeramos a continuación, reducirá la lista a unas pocas o, quizá, a una sola licencia.

45 *Una explicación detallada sobre cómo hacerlo correctamente se encuentra en: http://www.softwarefreedom.org/resources/2007/gpl-non-gpl-collaboration.html.*





A) La importancia de usar una licencia reconocida.

La primera cuestión que debemos considerar (y la más esencial) es la conveniencia de no volver a escribir una licencia nueva específica para nuestro proyecto, sino seleccionar una del listado de licencias libres y de fuentes abiertas reconocidas.[46] La proliferación de licencias crea problemas graves de incompatibilidad y fragmentación del sector del SFA, hasta el punto que la OSI está tratando de reducir el número de licencias certificadas activas.[47]

Usar una licencia reconocida

La FSF recomienda el uso de la licencia GPL, ahora en su versión 3, para garantizar la libertad del software. De hecho, es una tendencia generalizada limitarse a usar las licencias más comunes (las licencias GPL, LGPL, BSD modificada, MPL, Apache, CPL, CDDL, etc.).

Una ventaja de las licencias de SFA reconocidas es que son estándares: cualquier usuario experimentado conocerá el alcance de sus derechos y obligaciones. Además, su uso facilita la integración de componentes en programas más complejos: una vez se han determinado las licencias compatibles con la del proyecto, ya no será necesario revisar en detalle las condiciones aplicables a componentes bajo las mismas.

Existe, por supuesto, la posibilidad de escribir nuevas licencias y hasta someterlas a la certificación por la OSI. Pero en contra de lo que sucede con el uso de una licencia conocida, cada nueva licencia complica la tarea de revisar la compatibilidad de los componentes a la hora de integrar programas.

B) Los objetivos de la liberación del software

Los objetivos de nuestro proyecto:

•¿Son puramente filosóficos? En este caso, o bien no nos importará lo que se haga posteriormente con el programa (y entonces escogeremos una licencia permisiva), o al contrario, se exigirá una licencia con copyleft fuerte.

•¿Está prevista la provisión de soluciones comerciales basadas en este software una vez liberado? Una licencia mixta podrá ayudar a crear una comunidad profesional para mejorar el código.

•¿El software es parte de un programa universitario o el resultado de una investigación? En este caso se deberán tener en cuenta los criterios relacionados con la difusión del conocimiento.

•¿Es parte de una iniciativa pública? Si la respuesta es afirmativa, pueden surgir razones vinculadas con una política determinada de reutilización de recursos, su integración en un plan de interoperabilidad nacional o internacional, o el fomento de la industria local.

46 En http://opensource.org/licenses o en http://www.gnu.org/licenses/license-list.html.
47 Ver en http://www.opensource.org/proliferation.





C) Compatibilidad entre componentes

En el caso de un software que ya incorpora (o se prevé que incorpore) componentes de SFA, **las licencias de estos componentes podrán determinar la licencia del proyecto:** es necesario revisarlas para identificar la lista de licencias compatibles, entre las cuales se deberá seleccionar la del proyecto[48]. No es lógico proponer una licencia permisiva para el software si un componente fundamental integrado en el mismo tiene una licencia con copyleft. De forma recíproca, no es razonable elegir la GPL si el software debe integrarse con un programa bajo una licencia incompatible con ésta, como son la MPL, la CPL o la CDDL.

GONG

En el caso de la herramienta GONG, liberada por CENATIC en el 2008, la selección de la licencia fue determinada por uno de los componentes principales del software. La aplicación se basa en gran parte en Alfresco DMS, un software libre de gestión documental distribuido bajo la licencia GPL.

D) El tipo de software que va a crear y distribuir el proyecto.

En general, y desde una perspectiva jurídica, es importante considerar si el software se va a usar sin modificaciones o, si por el contrario es más probable que se modifique y se integre en sistemas o soluciones más complejas. En este caso, seleccionaremos la licencia según el grado de permisividad o libertad que se quiera otorgar a terceros para la distribución de aplicaciones que integren o modifiquen el software.

Usos del software

• **Software destinado al usuario final:** no es habitual que las funciones de este tipo de software (Mozilla, OpenOffice.org, GIMP, CLAM[49]) sean modificadas para crear obras derivadas. Sin embargo, podría ser interesante permitir a terceros la posibilidad de integrar componentes, extensiones, etc. al producto.

• Software que es parte de un **sistema operativo**, un **framework o middleware**: en este caso es más probable que terceros quieran integrar su código en el programa. ¿Qué condiciones queremos imponer a la redistribución de nuevos productos integrados en el proyecto?

• **Librerías y componentes funcionales:** para la mayor difusión y uso de software que se integrará, con toda probabilidad, en otros programas, ofreciendo nuevas funcionalidades, una licencia permisiva (o con copyleft suave, como la LGPL o la GPL con la excepción "Classpath") podría ser la más adecuada.

48 Ver la cuestión relacionada con la compatibilidad de licencias.expuesta anteriormente en este capítulo.
49 Respectivamente, www.mozilla.org, www.openoffice.org, www.gimp.org y www.clamwin.com.





E) Seleccionar la licencia más adecuada para nuestro proyecto

Una vez comprendidas estas cuestiones previas a la selección de la licencia es necesario identificar aquella que cumple los criterios de nuestro proyecto.

El principal criterio de diferenciación de las licencias SFA es la existencia o no de pactos de copyleft y el grado de reciprocidad exigido por los mismos.[50]

Una licencia *copyleft* tiene varias consecuencias: no permitirá privatizar el programa y facilitará la publicación de modificaciones y extensiones, pero será incompatible con software bajo otras licencias de tipo copyleft.

Por contra, una licencia permisiva permitirá la incorporación en otros programas bajo cualquier tipo de licencia, pero también autorizará la privatización del software o la creación de *forks*[51]: distintas versiones del mismo programa original, distribuidos bajo cualquier tipo de licencia.

Otros criterios relevantes a la hora de seleccionar licencia

Existen otras cuestiones que han llevado a los licenciantes a incorporar pactos o crear nuevas licencias que den respuesta a las necesidades de los autores de SFA.. Estas son algunas de ellas:

• **El control de marca:** se han redactado licencias con pactos referentes al uso de la marca del proyecto, impidiéndolo u obligándolo.

• **Licencias de patentes:** la segunda y tercera generación de licencias (a partir de la MPL 1.0) incluyen pactos relativos a las patentes. En un entorno proclive a las patentes (en EE.UU. por ejemplo) será preferible usar una de estas licencias.[52]

• **Posibilidad de usar varias licencias:** en algunos casos se permite distribuir código bajo varias licencias distintas (licencias duales o múltiples), por ejemplo para solventar necesidades de compatibilidad entre aplicaciones. La más conocida es la MPL 1.1, que permite usar la MPL u otras licencias alternativas indicadas por el autor original.

• **Prestación de servicios remotos:** existen una serie de licencias modernas[53] que extienden el efecto de copyleft al software utilizado para ofrecer servicios web (tipo ASP), un tema ya comentado en el Capítulo 2, sobre la licencia Affero GPL.

• **Marco jurídico:** la Unión Europea creó la EUPL para tener una licencia más adecuada al marco legal europeo.

50 En el Capítulo 2 hemos hecho un recorrido por los diferentes tipos de licencia de SFA reconocidas, con sus características particulares: licencias permisivas, con copyleft fuerte o suave.
51 Proyectos paralelos que distribuyen una variación del código original. Por ejemplo, los sistemas Unix, FreeBSD, NetBSD y OpenBSD; o Mambo y Joomla (gestores de contenidos web).
52 Por ejemplo, las licencias MPL 1.1, CPL/EPL 1.0, Apache 2.0, CDDL 1.0, OSL 3.0, CPAL o GPLv3.
53 Las licencias Real Networks PSL, Apple 2.0, OSL 3.0, CDDL 1.0 y AfferoGPL en particular.





F) Escoger más de una licencia

Finalmente, existe la posibilidad de usar varias licencias bajo un **régimen o esquema de licencia dual o múltiple**, tal y como comentamos a continuación.

3.2.3. Régimen de licencia dual

Una de las opciones interesantes que nos ofrece el uso de licencias de SFA (dado que no son exclusivas) es la posibilidad de usar varias licencias para un mismo software, lo que se denomina régimen de licencia dual[54]. Esto permite ajustar la licencia a diferentes tipos de usuario y usos.

MySQL

El ejemplo clásico de uso del sistema de licencia dual es la base de datos MySQL[55]. Por un lado, ésta se distribuye bajo la licencia GPL a usuarios finales e intermediarios que distribuyen su software bajo licencias libres (integrando MySQL en sus productos). Por el otro, se ofrece bajo licencia propietaria a intermediarios o integradores que venden su software a usuarios finales bajo este tipo de licencias.

Esta estrategia es la usada sobre todo por empresas comerciales involucradas en el desarrollo y distribución de SFA (como MySQL, Trolltech, SugarCRM, entre otras) porque les permite:

• Participar y beneficiarse de los desarrollos y avances de comunidad libre, adquiriendo una versión del programa bajo licencia libre, y

•obtener ingresos a partir de la distribución del mismo software bajo licencia propietaria y comercial.

En este caso nos encontramos ante el principio quid pro quo: la empresa ofrece software a la comunidad (la versión libre) y recibe algo a cambio (correcciones de errores, extensiones, etc.).

Modelos de licencias duales

En el ámbito del SFA, se pueden distinguir varios modelos de licencia dual:

54 *Traducción del inglés "dual licensing".*
55 *Ahora parte del portafolio de Sun Microsystems Inc. En www.mysql.com.*





Modelo de licencia dual	Descripción	Ejemplos
Un solo software, dos licencias	La empresa titular ofrece el mismo software bajo licencia libre gratuita para usos libres (normalmente la licencia GPL), y bajo licencia privativa y de pago para su redistribución bajo este mismo tipo de licencias. El modelo de negocio de la empresa se basa en ingresos obtenidos a partir de la versión comercial.	MySQL, Trolltech, Jahia, Sleepycat, eXoplatform
Dos programas similares, dos licencias	La empresa ofrece: 1.una versión básica del programa bajo licencia libre y gratuita; y 2.una versión más completa bajo licencia privativa (para evitar redistribuciones) y de pago (para obtener ingresos). Las condiciones restrictivas de la licencia privativa impiden que los elementos adicionales de la versión más completa del software se liberen a la comunidad y puedan integrarse en la versión libre.	SugarCRM, Jahia, otros - un modelo similar al freeware
Un software, una licencia (pero con opción de cambiar a otra licencia)	El licenciante ofrece el software bajo términos de una licencia que permite el uso del software bajo otra/s licencia/s. La LGPL, por ejemplo, permite al usuario elegir usar el software bajo la GPL.	MPL CeCILL LGPL (EUPL)





3.3. VALIDEZ DE LAS LICENCIAS DE SFA Y OTRAS CUESTIONES JURÍDICAS

En este apartado trataremos brevemente algunos temas relacionados con las licencias de SFA de carácter puramente jurídico, pero no por eso, menos importantes.
Es importante resaltar que los aspectos tratados a continuación (sobretodo los relativos a la creación de derechos y obligaciones vinculantes y las garantías y responsabilidades del proveedor) se aplican de la misma forma (o de forma muy similar) tanto al SFA como al software privativo. Asimismo, son cuestiones cuya resolución es a menudo complicada, por lo que nuestros comentarios son meramente introductorios.

3.3.1. La cesión de derechos bajo licencia de SFA ¿es una renuncia?

Es habitual que surjan dudas sobre si el uso de licencias libres implica la renuncia de derechos por parte del titular. **La respuesta a esta pregunta es simple y clara: en ningún caso.**

Desde un punto de vista legal, una licencia de SFA es una cesión no exclusiva de algunos de los derechos del titular bajo ciertas condiciones. No es una venta (de la obra, como si fuera un objeto físico), ni una cesión total y en exclusiva de los derechos de autor a favor de un tercero, ni una renuncia a emprender acciones legales si un tercero incumpliera cualquier de las condiciones de distribución del SFA.

Como las licencias de SFA no son exclusivas, el licenciante (titular) retiene sus derechos sobre la obra. En particular, retiene su propio derecho a reproducir, transformar, distribuir y comunicar públicamente el software. Podría también volver a autorizar a terceros a explotarlo bajo otra licencia (un régimen de licencia dual). Y, por ejemplo, si el licenciatario no respeta las condiciones de la licencia, el titular/licenciante podrá resolver la licencia (y así prohibir a esa persona que use el programa) o instar al licenciatario a que cumpla con sus obligaciones.

3.3.2. ¿Las licencias SFA crean obligaciones vinculantes, a modo de contrato?

La licencia de software, como cualquier contrato, requiere la aceptación manifiesta de una oferta válida: cuando una persona acepta las condiciones de uso de un software se convierte en un usuario legítimo y adquiere el derecho a usarlo.

Sin embargo, las licencias de software (salvo aquéllas de software más complejo, destinado a empresas) no se suelen plasmar a modo tradicional: en papel y firmado por las





partes. El software se distribuye, bien a través de Internet, bien mediante copias físicas (CD-ROM, DVD), difiriendo la aceptación de la licencia a un momento posterior a la compra del soporte. Cabe destacar que, actualmente, los sistemas habituales de aceptación – shrink-wrap, click-wrap o browser-wrap[56] - están generalmente reconocidos a la hora de establecer la existencia de un contrato, aunque puedan presentar algunas dificultades.[57]

Protección de los usuarios

Precisamente, como las licencias son contratos redactados unilateralmente, la ley impone una serie requisitos cautelares para asegurar que el usuario ha tenido oportunidad de acceder a las condiciones contractuales y que las ha aceptado. Además, la ley intenta proteger a la parte que no las ha redactado e imposibilita la validez de aquellas cláusulas que pudieran considerarse abusivas.[58]

Pero la mayoría del SFA ni siquiera utiliza estos procesos, sobretodo para componentes que no son programas instalables, como librerías u otros tipos de programas que se integran en algo más amplio. Su descarga de Internet (Sourceforge o el sitio de un proyecto particular) y hasta su integración dentro de software redistribuido por el licenciatario no siempre presenta ni requiere aceptación expresa de licencia alguna, y a veces ni siquiera se indica la licencia aplicable. Consideramos que éste es un proceso insuficiente para suponer la aceptación de las condiciones por parte del usuario, de acuerdo con nuestra tradición jurídica.

Aceptación de una licencia SFA

Ante este problema se argumenta que las licencias son meras autorizaciones bajo el Derecho de la Propiedad Intelectual: siempre que el usuario respete las condiciones de uso, gozará de los derechos especificados en la licencia. La GPL, en la tradición anglosajona, argumenta que esta licencia se convierte en contrato vinculante cuando el usuario modifica y redistribuye el programa porque, para ello, habrá accedido al código fuente, dónde reside normalmente la licencia. Por esta razón la modificación y posterior redistribución podrá considerarse una manifestación explícita de aceptación de las condiciones la licencia.

Desde la perspectiva jurídica española este es un tema pendiente de resolver, ya que será difícil afirmar y/o demostrar que el usuario realmente ha aceptado las condiciones establecidas por la licencia.

56 La rotura del precinto del paquete o caja en el que vienen los CD-ROM o DVD con el software (la licencia denominada shrink-wrap); la aceptación expresa de una licencia durante el proceso de descarga o instalación de un software (la licencia click-wrap); o la publicación de la licencia en Internet, con una referencia por medio de un hipervínculo desde el lugar de descarga (una licencia browser-wrap).

57 Siempre y cuando se hayan hecho las advertencias previas y se haya dado al potencial licenciatario la posibilidad de devolver el software y el contrato.

58 Real Decreto 1906/1999, de 17 de diciembre, por el que se regula la contratación telefónica o electrónica con condiciones generales. Ley 7/1998, de 13 de abril, de condiciones generales de la contratación. Real Decreto Legislativo 1/2007, de 16 de noviembre, por el que se aprueba el texto refundido de la Ley General para la Defensa de los Consumidores y Usuarios y otras leyes complementarias.





Asegurar un contrato vinculante

Para evitar posibles conflictos legales se recomienda establecer un procedimiento transparente e informativo que consista, básicamente, en indicar la licencia aplicable (en el CD de distribución, en el sitio web del proyecto o incluso en otra web, mediante un enlace de hipertexto)[59] a la hora de distribuir el software, con el objetivo de incorporar las condiciones de la licencia en el contrato entre titular y usuario.[60] De la misma forma, también es recomendable que un software de usuario final incluya, en su procedimiento de descarga o ejecución, la presentación y aceptación de la licencia.

3.3.3. ¿Las condiciones establecidas por las licencias de SFA son válidas?

Una vez establecido un contrato de licencia de fuentes abiertas entre las partes, no deberían surgir problemas con respecto a la validez legal de sus términos como licencia de software (cesión de derechos, imposición de obligaciones, resolución de la licencia, obligaciones sobre la publicidad o el uso de marcas).

Lo anterior es sin perjuicio de la existencia de cláusulas particulares que podrían considerarse nulas desde el punto de vista jurídico, según cada caso y según el derecho aplicable a la licencia. Nuestro derecho imperativo (de obligado cumplimiento) se preocupa de que las licencias que se hayan redactado unilateralmente y que puedan contener cláusulas abusivas no vinculen, total o parcialmente, a la parte más débil o pasiva de la contratación.

El régimen de protección de los consumidores

No serán válidos aquellos pactos de las licencias SFA que infringen las obligaciones imperativas de la protección del consumidor y del usuario (garantías de conformidad, por ejemplo), por falta de claridad y comprensión de las condiciones aplicables al producto comercializado (debido a ciertas exclusiones que no tienen equivalente jurídico en nuestra legislación), o por no ser inteligibles las advertencias mínimas que se imponen.

Todo ello, sin perjuicio de que el SFA sea habitualmente gratuito y su distribución pueda considerarse incluso como una donación, fuera del ámbito de la protección del consumidor.

Algunos de los pactos que pueden dar lugar a confusiones y ser considerados nulos jurídicamente son:

- Las limitaciones de garantías y responsabilidades, que suelen ir más allá de lo permitido por nuestro marco jurídico.

59 Proceso aceptado por tribunales europeos en varias ocasiones, por ejemplo en el caso Netfilter (Tribunal del distrito de Munich, 19/05/2004)

60 Si bien, en función de cada país, suelen existir particularidades respecto a lo que se considera suficiente y razonable" para lograr ese efecto.





- La determinación del foro jurisdiccional competente para resolver conflictos relacionados con el consumidor (p.e. la licencia MPL establece los tribunales de California y la CPL, los de Nueva York). Nuestra ley permite a los consumidores iniciar acciones legales y ser demandados ante los tribunales de su domicilio.

El hecho de que una licencia esté redactada en inglés no será, en la mayoría de los casos, causa de invalidez del pacto establecido entre un licenciante y un licenciatario profesionales.[61] En todo caso, si las partes llevan un conflicto a los tribunales españoles, podrán presentar una traducción jurada de la licencia.

Traducciones

Es fundamental usar una sola versión lingüística de las licencias para facilitar la compatibilidad entre ellas, porque sólo de esta forma podrán integrarse y redistribuirse dos componentes bajo condiciones de copyleft. Cualquier error o diferencia de traducción hará que los dos componentes se distribuyan con dos licencias diferentes, a no ser que la licencia incorpore un mecanismo de "compatibilidad entre versiones lingüísticas", como es el caso de las licencias la EUPL 1.1 para el software o Creative Commons v 3.0 para textos y otras obras.

3.3.4. ¿Cómo hacer cumplir las obligaciones de la licencia?

Ya hemos visto que el uso de una licencia de SFA no implica la renuncia de los derechos de propiedad intelectual. Por lo tanto, **el titular mantiene los derechos que le permitirán iniciar acciones legales contra las personas que infrinjan las condiciones de la licencia, para exigir su cumplimiento y en su caso reclamar una indemnización.**

Sentencias

En España, fuera de algunas sentencias relativas a licencias Creative Commons, no ha habido hasta el momento casos de mención en relación a licencias de SFA.

Pero consideramos interesante hacer referencia al caso Netfilters, en Alemania, en el que el titular del software, distribuido bajo la GPL, ha logrado hacer cumplir ante los tribunales alemanes, y en cuatro ocasiones, las condiciones establecidas por la licencia. No sólo ha conseguido el cumplimiento de sus términos (en este caso, la entrega del código fuente y la incorporación del texto de la licencia en la documentación del producto infractor) sino también una indemnización correspondiente a los costes legales incurridos y el pago de daños morales a una organización sin ánimo de lucro.[62]

61 Otra cosa será que muchas de las cláusulas, escritas normalmente bajo terminología propia de los EEUU y sin paralelismo o equivalente español, se puedan considerar oscuras o abusivas: por no entenderse en su significado o por ser contrarias a las normas españolas de obligado cumplimiento, respectivamente, se considerarán nulas y como no puestas en el contrato de adhesión.

62 Para detalles, ver en www.gpl-violations.org





En este sentido, son varias las situaciones que podrían ser causa de conflicto. Por ejemplo, el incumplimiento de las condiciones de una licencia (en particular, las obligaciones sobre la redistribución del software), divergencias respecto a la interpretación y aplicación de la condiciones de una licencia, o conflictos relativos a la titularidad de un software.

Infracciones típicas de la GPL:

- Integrar y redistribuir componentes bajo licencias incompatibles (en un software compuesto).
- No incluir una copia de la licencia con el software distribuido.
- No proporcionar o dar acceso al código fuente del software.
- No incluir los scripts de compilación e instalación con el código fuente de software bajo GPL.

En la práctica, el proceso habitual para hacer cumplir las obligaciones de las licencias SFA es extra-judicial. Normalmente, que el titular de los derechos sobre el software envíe una comunicación -un requerimiento formal- a la persona que incumple las condiciones de la licencia será suficiente para que la parte infractora tome las medidas adecuadas para remediar la situación.

En todo caso, siempre será posible iniciar un recurso ante los tribunales (una vez determinado ante qué tribunales deben iniciarse las acciones legales).

3.3.5 Garantías y responsabilidades

Como para cualquier software que se distribuya en el mercado – privativo o libre - surge la cuestión de las garantías que debe o no dar el proveedor y sus potenciales responsabilidades.

Garantías

El Derecho español establece unas mínimas garantías legales a favor de los que adquieren bienes y servicios. Estas garantías varían en cada caso pero, básicamente, son las de la conformidad del producto (el buen funcionamiento del software) y la obligación de subsanar defectos ocultos.

Nuestras leyes permiten regular, en cierto grado y dependiendo de la relación entre las partes -sobre todo entre profesionales- el derecho de reclamación del usuario (comprador) en caso de no conformidad. En este contexto, las licencias de SFA intentan excluir cualquier garantía con la formula "AS IS" ("tal cual"), y proporcionar el software "sin garantía alguna".





Esto podría justificarse (legalmente) en base a que el software se licencia de forma gratuita. Se podría considerar el SFA como una donación, para la cual no habría garantías (aunque sí responsabilidades, como indicaremos a continuación). Sin embargo, un tribunal podría considerar que estas limitaciones son abusivas para el usuario final y, por lo tanto, inválidas.[63]

Al margen de ello, es posible que un proveedor de soluciones basadas en SFA tenga que dar garantías respecto de su producto y/o servicio. Aunque no sea el licenciante del SFA, el cliente es comprador de servicios de programación e instalación y se beneficiará de unas mínimas garantías en relación a estos servicios, siempre que no estén válidamente limitadas en el contrato.

Responsabilidades

De manera similar a las garantías, nuestro sistema jurídico establece que un proveedor debe responder de los daños causados por sus productos, por ejemplo a causa de un error en el programa entregado o el servicio prestado. Sin embargo, en determinadas circunstancias, permite limitar estas responsabilidades.

Todas las licencias de fuentes abiertas establecen una limitación de responsabilidades en los disclaimers o "pactos de limitación de responsabilidades". Ante la posible invalidez de estos disclaimers (según cada marco legal), se suele acotar la limitación "en la medida permitida por *derecho aplicable*".[64] Ello significa que la licencia no excluirá las responsabilidades impuestas por nuestro derecho imperativo. En el caso español, el proveedor siempre deberá responder por haber causado daños corporales o la muerte de una persona, y no podrá limitar sus responsabilidades en casos de dolo y, según la situación, casos de culpa o negligencia.[65]

Por otra parte, estas limitaciones nos serán válidas ante los consumidores pero sí entre profesionales, siempre que la cláusula no sea considerada injusta y abusiva, ni vulnere la buena fe, dadas las circunstancias definidas por contrato.

En todo caso, la cuestión de garantías y responsabilidades no debe ser un motivo de preocupación si la distribución del SFA se ha gestionado adecuadamente y, en caso de distribuciones comerciales, provisión hecha en el contrato de suministro e implantación.

63 Nuestras leyes otorgan protección especial para los consumidores que el contrato de licencia y/o distribución no podrá limitar.

64 Esta acotación, que es válida en España (entendiéndose nulo sólo aquello que vaya en contra de las normas de obligado cumplimiento), y también habitual en otros países, no es en cambio válida ni posible en algunos otros países en los que se entendería todo el contrato nulo.

65 Art. 1101 Código civil y siguientes.





CAPÍTULO 4

EL USUARIO DE SOFTWARE DE FUENTES ABIERTAS





Ficha resumen

OBJETIVO	Presentar las principales cuestiones jurídicas que surgen a la hora de usar SFA.
DESTINATARIOS	Usuarios de software bajo licencia SFA de todo tipo: individuos, empresas, Administración Pública.
RESUMEN	Los usuarios de SFA se benefician de los derechos fundamentales del software libre y de fuentes abiertas: los derechos de uso sin discriminación, de reproducción, modificación, comunicación pública y distribución. Las licencias de SFA permiten: • A los individuos: instalar y usar el software sin discriminación, así como compartirlo con terceros. • A las empresas: una amplia libertad para adaptar el software a sus necesidades y a sus procesos de negocio, además de la independencia frente al proveedor y una garantía de continuidad tecnológica (al tener acceso al código fuente). Asimismo, las empresas pueden difundir el software sin ningún coste añadido sin perjuicio de las obligaciones de copyleft (en el caso de usar SFA bajo copyleft) cuando distribuyen el software fuera de la organización. • A las administraciones públicas: además de los beneficios mencionados para empresas, el SFA les permite ganar en eficiencia, cumplir con el marco legal de reutilización de recursos digitales, y fomentar la industria local de servicios informáticos.
REFERENCIAS	En la Guía: - Capítulo 3 sobre aspectos prácticos de las licencias SFA para usuarios finales - Capítulo 5 sobre liberación de proyectos - Capítulo 6 sobre servicios de integración y soporte basados en SFA En la red: - Informes de la UE (AAPP): *http://www.osor.eu/expert-studies* - Informe del CENATIC sobre el SFA en la Administración Pública española: *http://observatorio.cenatic.es/index.php?option=com_rubberdoc&view=doc&id=38&format=raw*





4.1. EL USO DE SFA POR PARTE DE LAS ORGANIZACIONES E INDIVIDUOS

Este capítulo versa sobre las implicaciones jurídicas que puede tener el uso del SFA para los usuarios finales, según la siguiente clasificación:

• **El usuario individual** suele ser mero usuario, es decir, que sus aplicaciones son de escritorio (ofimática, navegador, correo electrónico, editor de imágenes, etc.) y aunque pueda llegar a redistribuir el software a sus amigo s y conocidos, no suele modificarlo.

• **Las empresas u otras organizaciones con departamento de tecnología** son usuarias cada vez más intensivas de SFA. Su caso es más complejo, ya que no solamente usan el SFA, sino que lo adaptan y lo integran con otro software y pueden volver a distribuirlo, tanto dentro como fuera de la entidad o grupo empresarial.

Al igual que las empresas, **las Administraciones Públicas** adoptan de manera creciente el SFA. Su papel en la economía y las reglas que rigen su actuación las convierten en un usuario particular.

Usuario y colaborador

El usuario final de SFA (tanto el consumidor individual como las organizaciones) adquiere un papel que va más allá de ser meramente el beneficiario del software. Puede participar activamente en su desarrollo, enviando comentarios y requerimientos al proyecto, haciendo pruebas, informando sobre bugs y hasta proporcionando correcciones de errores. De esta manera, su participación adquiere gran importancia en el círculo de colaboración que fomenta el uso de SFA. Si bien esto no es lo más común cuando hablamos de usuarios individuales, los departamentos de tecnología de las organizaciones sí contribuyen de forma relevante en los proyectos de SFA.

4.2. ASPECTOS COMUNES

Antes de presentar las implicaciones legales específicas para cada tipo de usuario, es conveniente hacer repaso de aquellos aspectos jurídicos que tienen en común: la libertad de uso, la independencia frente al proveedor y las patentes de software.

4.2.1. Las libertades de uso

El primer aspecto, y el más importante para cualquier usuario, es que todos nos beneficiamos de los derechos fundamentales que conceden las licencias de SFA: el derecho a copiar el software cuantas veces sea necesario, a modificarlo según nuestras necesidades o adaptarlo a sistemas existentes, y a redistribuir el programa internamente o a terceros.[66] El acceso al código fuente permite al usuario ejercer estos derechos, además de estudiarlo si se quiere entender cómo funciona.

66 Usamos aquí términos no jurídicos: copiar corresponde al concepto jurídico de la "reproducción", modificar a la "transformación" y redistribuir corresponde a "distribución" y "comunicación pública" (ver el Capítulo 7).





Veamos cada uno de estos aspectos con más detalle:

• **El uso[67]:** es condición fundamental de las licencias de SFA que permitan a cualquier usuario hacer "lo que quiere" con el software, sin discriminación ni restricción. De hecho, se ha denegado la condición de fuentes abiertas a licencias con restricciones sobre usos militares o nucleares, o sobre países particulares.

La no discriminación

Recordemos dos criterios de la Definición de Software de Fuentes Abiertas:

1. La no discriminación con respecto a las personas o grupos: la licencia no debe discriminar a ninguna persona o grupo de personas.

2. La no discriminación con respecto a los sectores de actividad: la licencia no debe restringir el uso del programa a sectores de actividad específicos. Por ejemplo, no puede impedir que el programa sea usado en un negocio particular o para la investigación genética, por ejemplo.

• **La copia:** el derecho de "reproducción" o de copiar el software, no solamente permite instalarlo en cuantos equipos consideremos necesarios (por ejemplo, en todos los PC's de una empresa), sino que también es fundamental para poder modificar el programa y redistribuirlo.

• **La modificación:** el derecho de modificar el SFA es necesario para poder adaptarlo a nuestras necesidades. Mientras que un usuario consumidor no suele modificar sus programas, las organizaciones se beneficiarán intensamente de este derecho para adaptar el software a sus procesos o modelos de negocio, integrarlo con otras aplicaciones, analizar su estructura y funcionamiento, verificar su nivel de seguridad, etc.

¿Qué se puede hacer con el software de fuentes abiertas?

• Las licencias de SFA permiten realizar, entre otras, las siguientes acciones:

• Descargar, instalar y ejecutar el SFA sin limitaciones: en un ordenador, en varios, en una red.

• Hacer una copia de seguridad del software.

• Descargar el código fuente y estudiarlo.

• Analizar las interfaces para hacer interoperable un software.

• Modificar el software para adaptarlo a sus necesidades, recompilarlo y ejecutarlo.

• Utilizar parte del código para desarrollar otro software.

• Mejorar y ampliar el software original con nuevas funcionalidades, extensiones y plug-ins.

67 Jurídicamente, la ley no contempla expresamente el uso como forma de explotación. Sin embargo, en la práctica, es un concepto recogido en la figura del usuario legítimo.





• Integrarlo en otro software (SFA) para mejorar sus funcionalidades.

• Redistribuir o comunicar públicamente el software original y sus modificaciones y extensiones en un CD, memoria USB, una página web, o en redes de pares P2P (respetando las condiciones de licencia sobre la redistribución).

• Crear documentación sobre el software y venderla.

• Según la licencia, incorporarlo o usarlo en una aplicación y distribuir ésta bajo licencia no libre.

• **La distribución y la comunicación pública:** estos derechos autorizan al usuario a compartir el SFA con otras personas, tanto la versión original como las modificaciones realizadas sobre la misma. Por ejemplo, un consumidor puede compartir un CD de Knoppix o OpenDisc con amigos[68]; una empresa puede distribuir un software basado en SFA a las sucursales y empresas afiliadas de su grupo; una Administración Pública puede poner un SFA a disposición de cualquier otra administración o ciudadano, a menudo a través de forjas con herramientas de colaboración y gestión del código[69].

4.2.2. La independencia frente al proveedor

El acceso al código fuente, con derechos de explotación (de copia, modificación, etc.) permite a los usuarios ser **independientes frente a sus proveedores:**

• Para poder arreglar un problema de un software privativo, el usuario debe contactar con el servicio de soporte técnico del fabricante del software. Si esta empresa deja de prestar soporte sobre una versión del software, el usuario no tiene más opciones para solucionar dudas o errores.

• En cambio, si tenemos problemas con un SFA (conflictos con el proveedor original, su desaparición o si éste deja de prestar servicios sobre este software), nada nos impide cambiar de proveedor o mantener y mejorar el software de manera interna.

4.2.3 Patentes de software

Finalmente, hacemos una breve referenca a las patentes de software. Recordando que su validez en el marco jurídico europeo es discutible, tampoco debemos negar su existencia y potencial impacto para el el usuario final. Comentamos este tema con más detalle en el Capítulo 8.

68 http://www.knoppix-es.org/ y http://www.theopendisc.com/

69 Se usa el término de "forja" para repositorios de SFA que ofrecen herramientas de gestión del código. Forjas públicas incluyen, entre otras, el repositorio de software libre de la Junta de Andalucía (http://www.juntadeandalucia.es/repositorio/) y Lafarga de la Generalitat de Cataluña (www.lafarga.cat) o del CENATIC (actualmente en http://cenatic.morfeo-project.org/)





4.3. ASPECTOS PARTICULARES

No todos los usuarios utilizan el SFA de la misma manera, por lo que las implicaciones legales para cada uno también son diferentes:

4.3.1. Individuos y SFA

Desde la perspectiva de los derechos de autor, los usuarios individuales de SFA gozan de todos los derechos otorgados por sus respectivas licencias. Como ya se ha comentado, no suelen existir condiciones o restricciones para el mero uso de software de fuentes abiertas. Al no modificar ni redistribuir los programas, el usuario individual no se ve afectado por las condiciones y obligaciones que las licencias imponen a la redistribución del programa[70], sobretodo por el efecto *copyleft* (que sí deberá tener en cuenta el responsable de TI de una organización).

En este contexto, a menudo surgen dudas respecto a las garantías y responsabilidades derivadas del uso de este tipo de software. La mayoría de las licencias de SFA (y también de software privativo estándar) ceden los derechos de uso del programa sin garantías y limitan las eventuales responsabilidades de su proveedor (las exclusiones o *disclaimers*)[71], lo que nos lleva a pensar que el consumidor está desprotegido. Mientras que la validez de las limitaciones de responsabilidades ante un consumidor es discutible, la exclusión de garantías podría defenderse desde la perspectiva del proyecto o proveedor del SFA en la medida en que el software se proporciona de manera gratuita y, por lo tanto, fuera de las relaciones comerciales, en las que sí aplicarían las garantías previstas por ley.

Aún así, como ya hemos comentado en el Capítulo 3, el usuario de SFA no quedará desprotegido. Bajo las leyes de protección del consumidor, algunas cláusulas – las limitaciones de responsabilidades en particular – podrían considerarse abusivas y, por lo tanto, nulas. En este caso, el individuo gozará de los derechos y protección proporcionados por el marco jurídico imperativo de la protección del consumidor.



70 Ver apartado 34.2 abajo.

71 A modo de comparación, estas limitaciones son muy similares a las incluidas en las licencias propietarias sobre software estándar de usuario final – por lo tanto el usuario de SFA no está en una posición muy diferente en relación con el software privativo.





4.3.2. Empresas y organizaciones

El uso de software de fuentes abiertas se está incrementando en las organizaciones privadas y públicas: se llevan a cabo migraciones de servidores y estaciones de trabajo a GNU/Linux y se instala cada vez más SFA como cortafuegos, sistemas de seguridad, gestores de contenidos para páginas web, nuevas aplicaciones de escritorio o de front office, aplicaciones de tramitación en línea, etc.

Uso desconocido de SFA

Además de un uso explícito o planificado de aplicaciones de SFA, la empresa puede tener instalado y utilizar más SFA del que piensa, debido a:

• La incorporación de SFA en proyectos informáticos sin el conocimiento de los responsables del proyecto (internamente o por parte de proveedores).

• La herencia de tecnologías basadas en SFA en fusiones o adquisiciones de empresas.

• La adquisición de productos que incorporan SFA (entre otros, dispositivos móviles y de red, cortafuegos, etc.).

Dentro de las organizaciones, se puede adoptar SFA de varias maneras y con distintos grados de intensidad e impacto jurídico , tal y como mostramos en la siguiente gráfica:

Modelo de uso de SFA en la empresa

Creación proactiva de valor
IMPACTO
Uso pasivo de las tecnologías
5. LÍDER
4. COLABORADOR
3. SPONSOR
2. CONTRIBUIDOR
1. USUARIO
0. NEGACIÓN
Un sólo producto
ALCANCE
Múltiples proyectos


Adaptado de: Peter Carbone (2006): Competing With Open Source Software: Insights From Recent Research.[72]

72 En http://www.ocri.ca/events/presentations/partnership/April2106/Carbone.pdf





Cuanto más intensivo sea el uso de SFA en la organización, más importante será su impacto jurídico, lo que ilustramos en la siguiente tabla. Comentaremos estos aspectos a continuación.

NIVEL			USO	ASPECTOS LEGALES (acumulativos)
USUARIO	1.a	**Usuario interesado**	Sustitución de aplicaciones propietarias de back-office. Instalación de aplicaciones de usuario (escritorio, back-office).	• EULA - derechos de uso. • Ausencia de garantías y limitaciones de responsabilidades. • Soporte de la comunidad.
USUARIO	1.b	**Usuario avanzado**	Uso y adaptación de SFA para aplicaciones empresariales más estratégicas. Infraestructura / ecosistema tecnológico de SFA.	• Garantía de uso pacífico por parte del proyecto / Indemnizaciones de propiedad industrial. • Garantías de funcionamiento y conformidad del proveedor. • Soporte y mantenimiento profesional. • Acceso al código fuente.
PARTICIPANTE	2.	**Contribuidor**	La empresa contribuye software al proyecto que gestiona las aplicaciones que utiliza.	• Cesión al proyecto de derechos sobre las contribuciones. • Confidencialidad (del software propio).
PARTICIPANTE	3.	**Sponsor**	La empresa patrocina y contribuye recursos al proyecto de SFA. Obtiene información y ventajas directas del proyecto.	• Participación en la gestión del proyecto. • Confidencialidad (software del proyecto).





NIVEL			USO	ASPECTOS LEGALES (acumulativos)
PARTICIPANTE	4.	**Colaborador**	La empresa participa en la dirección del proyecto libre, orienta su evolución hacía sus necesidades.	• Gestión de propiedad y licencias con respecto al proyecto libre.
LÍDER	5.	**Líder**	La empresa lidera un proyecto liberado por ella.	(ver Capítulo 5 – Proyectos libres) • Gestión de propiedad, licencias. • Selección y uso de marcas. • Patentes.

Nivel 1.a: El usuario interesado

Por definición, una licencia libre y de fuentes abiertas no puede discriminar los diferentes usos que se hagan del programa, ni a los diferentes tipos de usuario: individuos, empresas, fundaciones, administraciones públicas, etc. Por lo tanto, **cualquier organización, como licenciataria del software, se beneficia de las libertades básicas otorgadas por las licencias de fuentes abiertas.**

Sin embargo, la empresa deberá tener en cuenta algunos factores relevantes en este sentido:

• El software se usa bajo una licencia que excluye garantías y limita las responsabilidades del proveedor (la validez de estas cláusulas será mucho más probable en este caso que ante consumidores).

• Su único soporte será a través de la comunidad del proyecto, que puede ser más o menos activa, y fuera del ámbito de control de la empresa.
La evolución independiente del proyecto no seguirá necesariamente las necesidades de la empresa.

Como ocurre con el software privativo, la empresa debería conocer y gestionar su uso de SFA, con el propósito de alinearlo con los objetivos y necesidades de la organización.





Una correcta gestión de este uso evitará incertidumbres y problemas no previstos (daños causados por software de baja calidad, requerimiento de cesación por haber infringido derechos de terceros, etc.)[73]

Infracciones de derechos de terceros

La empresas y organizaciones deberán tener en cuenta y valorar el riesgo de infringir derechos de terceros o cláusulas de licencias de software – un riesgo que existe tanto sino más en el ámbito privativo como para el SFA.

El SFA se crea en entornos abiertos, por muchos desarrolladores que trabajan en diferentes lugares y empresas, que proporcionan código de manera más o menos anónima (para el usuario) y lo proveen libre de garantías.[74] Por lo tanto, se argumenta que existe un mayor riesgo de que programas de terceros se incluyan dentro del programa usado por la organización, sin autorización o bajo una licencia incompatible con la licencia de distribución.

Podríamos decir que este riesgo es muy bajo en el caso de proyectos libres de reconocido prestigio (como los programas de la Fundación Apache, los programas GNU, Linux, OpenOffice.org) u otros proyectos que llevan una gestión muy detallada de la propiedad intelectual.

Sin embargo, éste sí es un aspecto a tener en cuenta cuando hablamos de proyectos más pequeños (menos visibles) que generan componentes que se integran en aplicaciones más complejas o en soluciones desarrolladas a medida.

Aún así, este riesgo debe matizarse: la visibilidad del código fuente permite verificar si el programa proviene de terceros y la comunidad está muy alerta a la hora de detectar infracciones de este tipo.[75]

Hasta ahora, no se conoce ningún software de fuentes abiertas que haya plagiado a otro programa. En cambio sí se han visto casos de incumplimiento de licencias de SFA (falta de entrega del código fuente, ausencia de una copia de la licencia en la distribución): los casos Netfilters en Alemania o, más recientemente, BusyBox en Estados Unidos, son ejemplos de ello.[76]

Nivel 1.b: El usuario avanzado

Cuando la empresa llega a ser usuaria avanzada de SFA, la dependencia del programa crece. En este caso es importante gestionar correctamente ciertos aspectos legales a la hora de proponer, diseñar, aprobar e implementar un proyecto basado en software de fuentes abiertas. Esta gestión incluye:

73 Como cualquier otro software. La diferencia en cuanto a SFA se refiere es queualquiera puede descargar y, eventualmente, instalar un SFA evitando entrar en el proceso de compra y gestión de licencias de la empresa.

74 En las empresas de software privativo el desarrollo es supuestamente más organizado y controlado.

75 Asimismo, empiezan a ofrecerse servicios de code screening (revisión de código) que permiten comparar el código fuente de una aplicación con una base de datos de SFA, para averiguar su origen y régimen de licencia: Black Duck protexIP en www.blackducksoftware.com, Palamida IP Amplifier en www.palamida.com.

76 Ver en www.gpl-violations.org y http://www.softwarefreedom.org/news/2008/mar/17/busybox-verizon/





• Exigir la entrega del código fuente del SFA y preservarlo, para tener la máxima garantía de continuidad.

•Licenciar versiones profesionales del SFA, con garantías en cuanto a la propiedad intelectual y el uso pacífico del producto.

• Contratar, con un proveedor experto o con el mismo proyecto, aquellos servicios relevantes para la calidad y continuidad del software (garantías de conformidad y funcionamiento, soporte y evolución del producto)[77].

En los contratos de suministro, será recomendable exigir el uso de estándares abiertos, permitiendo la interoperabilidad de los productos.

Esta gestión será igual de importante en el caso de que la empresa use software privativo en aplicaciones críticas para la organización.

Asimismo, la redistribución del SFA será un tema relevante a tener en cuenta, como comentaremos más adelante (apartado 5, copyleft y garantías). Finalmente, recordamos que la empresa tiene la posibilidad de participar de manera activa en la comunidad para mejorar el software e influenciar su evolución (y pasar al nivel 2 y siguientes).

Niveles 2 - 4: Progresar desde el nivel de contribuidor al de colaborador

En los siguientes niveles de actuación, la empresa deja de ser mera usuaria y empieza a colaborar con la comunidad, enviando contribuciones al proyecto (correcciones de errores, solicitudes de modificaciones, nuevas funcionalidades) para acabar participando en la definición de la evolución del mismo.

En estas circunstancias, surgen dos temas jurídicos importantes:

• La correcta cesión de contribuciones al proyecto: suele realizarse bajo un acuerdo sobre contribuciones o bajo la misma licencia del proyecto.

• La definición del ámbito de confidencialidad entre empresa y proyecto: determinando la información (y software) que debe mantenerse confidencial y lo que se puede compartir con el proyecto y la comunidad.

Líder

El último escalón en el uso de software de fuentes abiertas es cuando la empresa libera su propio software y gestiona el proyecto, hecho que implica varios aspectos jurídicos que comentamos con más detalle en el siguiente capítulo de esta Guía.

77 Ver el Capítulo 6 sobre la integración de soluciones basadas en SFA.





Conclusiones

Desde la perspectiva legal, el SFA ofrece muchas ventajas para los usuarios empresariales, si bien se recomienda implementar las mejores prácticas de gestión habituales en la adquisición e implementación de software: una evaluación de la madurez y reputación del software y del correspondiente proyecto, la búsqueda de un proveedor local sólido y la contratación de servicios de soporte según las necesidades del usuario y la obtención de garantías de la calidad legal de las soluciones adquiridas.

La necesidad de gestionar correctamente la adopción de SFA adquiere más relevancia a medida que este tipo de software ocupa un lugar más relevante en la infraestructura tecnológica de la empresa. Internamente esto puede requerir de un plan de formación de los responsables técnicos para mejorar los conocimientos de los aspectos jurídicos que afectan al software (tanto privativo como de fuentes abiertas), la identificación y revisión jurídica de los proyectos que incluyen SFA y la supervisión de las redistribuciones del software dentro y fuera de la organización.

La gestión de SFA en la empresa: las mejores prácticas

A continuación se listan las mejores prácticas que deberían implementar las empresas para beneficiarse al máximo de sus aportaciones a la comunidad de SFA. La mayoría de ellas también serán válidas para cualquier tipo de software, incluido el privativo:

• Reconocer internamente el valor e importancia del SFA, y del sector tecnológico en general.

• Revisar el uso interno de SFA, tanto explícito como implícito (dispositivos, tecnologías de terceros).

• Establecer, diseminar y mantener actualizada una política de uso del SFA.

• Establecer un centro de competencias o área de SFA para coordinar y promocionar su uso.

•Formar al personal relevante (responsables y gestores de TI, de innovación, de costes) en aspectos tanto técnicos como organizativos y legales.

Para empresas que hacen un uso más intensivo de SFA (en aplicaciones e infraestructura de TI) se recomienda establecer una política de uso del software, tanto para monitorizarlo y gestionarlo correctamente como para poder analizar las eficiencias y beneficios derivados de su implantación:

• Determinar la posición e integración del SFA en el proceso y estrategia de negocio de la empresa.

• Establecer un proceso de revisión y aprobación, liderado por un equipo mixto de abogados y técnicos.

• Definir y documentar la política y normas de uso del SFA y gestionar las licencias que correspondan (igual que en el caso de licencias privativas).





• Establecer procedimientos adecuados para la adquisición de SFA: obligaciones mínimas exigibles en contratos de suministro, revisión de vendedores y su calidad/ experiencia, modelos de contrato, criterios para casos de revisión del código.

• Definir reglas en caso de operaciones corporativas (fusiones, adquisiciones, ventas).

• Tener claras las reglas para participar en la comunidad (quién, cuándo, y cómo, y determinar qué se puede compartir y que no).

4.3.3. El sector público

El sector público en España, y también en Europa, es un gran consumidor o usuario de software. Pero también es creador, directa e indirectamente, a través de encargos y contratos públicos. El argumento de que el software "público" debe ser software libre y de fuentes abiertas toma cada vez más fuerza.

A nivel europeo, hay varios ejemplos de uso del SFA por parte de la administración pública. La migración de servidores de la administración de la ciudad de Munich es un caso conocido, pero existen muchos más. Ciudades y regiones instalan y liberan SFA. La Unión Europea establece, dentro de su programa Interoperable *Delivery of European eGovernment Services to Public Administrations, Businesses and Citizens*[78], un observatorio de SFA, que recoge casos de referencia de adopción de SFA en la U.E.[79]

Ventajas

Las ventajas del uso del software libre por la Administración Pública han sido discutidas en varios lugares. Destacamos la *Propuesta de recomendaciones a la Administración General del Estado sobre utilización del software libre y de fuentes abiertas*[80], MAP, de junio de 2005. Sintetizando, podemos enumerar las siguientes ventajas legales:

1. Obtener derechos suficientes sobre el software (control, por un lado y libertad, por el otro) para optimizar su gestión: actualizaciones, redistribuciones, etc.

2. Disfrutar de la libertad de copia y redistribución interna y externa a la Administración Pública.

3. Contribuir a la puesta en común y reutilización de software de las administraciones públicas (IDA, nacional).

4. Cumplir con el marco legal de la actuación pública (eficacia, eficiencia, conservación, seguridad, normalización e interoperabilidad, acceso y respeto lingüístico, reutilización de recursos).

78 En http://ec.europa.eu/idabc/ y http://osor.eu/.
79 Y que ha patrocinado la redacción de una licencia pública europea, comentada en el Capítulo 3 de esta Guía.
80 En http://www.csae.map.es/csi/pg5s44.htm.





Actuación del sector público

Queda claro que las Administraciones Públicas españolas son competentes para adquirir y utilizar SFA, intercambiarlo entre ellas y liberar software de su titularidad a la comunidad[81].

En cuanto a su adquisición, si bien ningún país europeo obliga a usar SFA, la mayoría prohíbe la discriminación de proveedores que presenten propuestas de SFA, basándose en los principios de libre competencia y no discriminación. Por contra, obligan a que el software suministrado cumpla con los estándares abiertos e internacionalmente reconocidos.

El marco de actuación del sector público en España

En España, existen algunas disposiciones generales que tienden a favorecer el uso de SFA en la Administración Pública, sin ser específicas[82]. La más relevante se encuentra en el Capítulo III del Titulo IV de la *Ley 11/2007 de 22 de junio, de Acceso Electrónico de los Ciudadanos a los Servicios Públicos (LACESP)*[83], titulado *"Reutilización de aplicaciones y transferencia de tecnologías"*, y en la disposición adicional decimosexta de la Ley 57/2007, de 28 de diciembre, de *Medidas de Impulso de la Sociedad de la Información (LISI)*[84].

LACESP: se establecen medidas que, sin promover directamente el SFA, facilitarán la reutilización de recursos informáticos entre administraciones, la creación de repositorios de aplicaciones para su reutilización y, eventualmente, la liberación de aquellas que son propiedad de las administraciones públicas bajo licencias SFA.

LISI: se establece que la AP puede poner a disposición del público aquellos contenidos digitales cuyos derechos de propiedad intelectual le pertenezcan sin restricciones o sean de dominio público, bajo licencias que permiten el estudio, copia y redistribución en los mismos términos (es decir, con un grado de copyleft). Notemos que no se habla de modificación, por lo que no se trata, necesariamente, de licencias libres.

81 *Ello, sin perjuicio de las cuestiones de responsabilidad en las que pueda incurrir la Administración ante los usuarios de dicho software una vez liberado.*

82 *Ley 30/1992 sobre el Régimen Jurídico y del Procedimiento Administrativo Común (obligación de transparencia de la actuación administrativa, eficacia y eficiencia); y RD 263/1996 sobre la utilización de técnicas electrónicas, informáticas y telemáticas (Seguridad y conservación de la información en apoyo electrónico, Normalización y Interoperabilidad).*

83 *En http://noticias.juridicas.com/base_datos/Admin/l11-2007.html*

84 *En http://noticias.juridicas.com/base_datos/Admin/l56-2007.html.*





El ejemplo más notable de liberación y uso de software es el repositorio de Andalucía, apoyado por la *Orden del 21 de febrero de 2005 sobre disponibilidad pública de los programas informáticos de la administración de la junta de Andalucía y de sus organismos autónomos*[85].

Las Administraciones Públicas europeas han empezado a publicar políticas de uso de SFA para promover su adopción, ayudar a sus administraciones locales a usarlo y hasta participar en el movimiento de fuentes abiertas.

Directrices e informes nacionales

Los gobiernos de Francia, Nueva Zelanda y Australia[86] han definido un proceso estratégico estándar para proyectos basados en SFA (con estándares de desarrollo, de migración y de liberación) en los que se identifican los roles y las personas responsables de los aspectos jurídicos del proyecto y cuáles deberían ser sus tareas. Se recomiendan, entre otras:

• Crear y hacer un seguimiento de documentación estándar tanto para el desarrollo como para el suministro de software basado en SFA: documentación sobre el programa, sobre los procesos de desarrollo y las decisiones de entrega de código, su liberación, etc.

• Definir e implementar un plan de formación para los desarrolladores y gestores internos de la Administración Pública.

Holanda ha establecido el *Programma Open Standaarden en Open Source Software voor de Overheid*[87], que ha publicado varias guías para favorecer la adopción de SFA en las Administraciones Públicas holandesas[88]. Lo mismo han hecho Alemania y la Comisión Europea.[89]

El gobierno español (el Ministerio de Administraciones Públicas) preparó el mencionado informe “Propuesta de recomendaciones a la Administración General del Estado sobre utilización del software libre y de fuentes abiertas” y el CENATIC ha publicado recientemente su “Informe Software de Fuentes Abiertas en la Administración Pública española”.[90]

Licencia para el software de las Administraciones Públicas

Además de ser usuario de SFA, la Administración Pública es proveedora: libera software bajo licencias de SFA ¿Qué licencia sería la más adecuada para esta liberación?

85 *Repositorio en http://www.juntadeandalucia.es/repositorio/, la Orden se encuentra en http://www.juntadeandalucia.es/ott/docs/lex/OR_21-02-06_SW-Libre-JA.pdf*

86 *Francia: Agence pour les Technologies de l'Information et de la Communication dans l'Administration. Y Adullact (Association des Développeurs et des Utilisateurs de Logiciels Libres pour l'Administration et les Collectivités Territoriales) en www.adullact.org/; Australia: http://www.sourceit.gov.au/sourceit/oss y http://www.agimo.gov.au/infrastructure/oss; Nueva Zelandia: http://www.e.govt.nz/policy/open-source/open-source-legal2/*

87 *Open Standards and Open Source Software (OSOSS), en http://www.ososs.nl/*

88 *Guidelines for the control of authority's risk with open source software. En http://www.ososs.nl/node/17778*

89 *Alemania: "Migration Guide v2, - A guide to migrating the basic software components on Server and workstation computers", de marzo del 2005 (no existen actualizaciones posteriores). U.E: Guidelines for Public Administrations on Partnering with Free Software Developers (2005) en http://osor.eu/expert-studies*

90 *Ministerio de Administraciones Públicas, en http://www.csi.map.es/csi/pg5s44.htm y http://www.cenatic.es/lang-es/publicaciones.html*





Dependerá de sus objetivos estratégicos. Actualmente, la mayoría del software liberado por las Administraciones Públicas españolas está licenciado bajo GPL. Después de un estudio jurídico, la Comisión Europea aprobó la redacción de una nueva licencia de fuentes abiertas para la liberación del software de la misma institución, publicada en enero del 2007: la European Union Public License, actualmente traducida a todos los idiomas oficiales de la UE.[91]

La EUPL

Todavía no hay muchos casos de uso de la EUPL, aunque se supone que será utilizada para liberar el software de la Comisión Europea[92]. Mientras tanto, en Europa, las Administraciones Públicas siguen publicando software bajo varias licencias existentes, a pesar de sus connotaciones americanas, de las cuestiones a tener en cuenta en relación a la validez de las exclusiones de garantías y responsabilidades, y del idioma utilizado (el inglés).

4.4. DOS TEMAS FUNDAMENTALES: EL COPYLEFT Y LAS GARANTÍAS SOBRE SFA

La obligación de publicar el código de sus programas y la ausencia de garantías y soporte para el SFA son dos preocupaciones habituales entre aquellos que deciden no usar software de fuentes abiertas. En este apartado, intentamos resolver las dudas que puedan existir al respecto.

4.4.1. El copyleft

Un tema que preocupa mucho a las organizaciones es el impacto del efecto *copyleft* y la obligación de publicar o distribuir el código fuente de sus programas. **Existe el mito de que las licencias de SFA obligan al usuario a publicar (libremente) cualquier modificación que se haga del software. Esta creencia es incorrecta.**

Hemos visto en los capítulos 2 y 3 de esta Guía que, si bien las licencias de SFA permisivas no imponen restricciones sobre el uso del software, las licencias de tipo copyleft, como la GPL, CPL u OSL, sí incluyen ciertas condiciones que es importante conocer y tener en cuenta. Estas condiciones incluyen la obligación de mantener los avisos de autor y de indicar las eventuales modificaciones, así como la de dar acceso al código fuente de los programas (modificados o no) que se distribuyan a terceros.

Esta última imposición puede causar ciertas inquietudes en una organización, sobre todo cuando los procesos o reglas del negocio están implementados en un programa. No debería ser así. **El *copyleft* obliga al usuario a distribuir u ofrecer acceso al código fuente de un programa solamente en el caso de redistribuirlo a terceros y, aun así, solamente**

91 En http://osor.eu/eupl. Comentada en el Capítulo 2 de esta Guía.

92 En OSOR, encontramos la lista en http://forge.osor.eu/softwaremap/trove_list.php?form_cat=307





al destinatario de la distribución (no al público, ni al titular original del software).[93] Por lo tanto, en ningún caso, la difusión interna (dentro de la entidad) de un SFA o su distribución a sus filiales/empresas del grupo, obligará a publicar el código fuente del software a terceros.

Casos de distribución no evidentes

Normalmente, la organización es la licenciataria final del software y no (re)distribuirá sus programas a terceros. Entonces no tendrá ninguna obligación de publicar el código fuente. Sin embargo, hay dos casos que podrían considerarse como redistribuciones y que no son evidentes:

1. La distribución o comunicación de un programa entre empresas subsidiarias y participadas de un grupo empresarial (o entre entes públicos), o en el caso de fusiones de empresas.

2. La distribución o comunicación del programa a empresas que prestan servicios informáticos de desarrollo o hosting de la aplicación a sus clientes.

En cada caso deberán tenerse en cuenta las condiciones establecidas por la licencia. La OSL y la CDDL no consideran que la distribución del software entre los miembros entre un grupo empresarial sea una "distribución" a efectos del copyleft. Y la licencia GPLv3, por ejemplo, excluye de las obligaciones de copyleft la transferencia del programa modificado a un proveedor a efectos de llevar a cabo un encargo o la prestación de un servicio.

Ejemplo: Checklist para usuarios de software bajo la GPL

Las licencias no imponen condiciones a los usuarios que instalan y usan SFA bajo la GPL. Sin embargo, para beneficiarse de los derechos otorgados por las mismas, así como para establecer las bases para un correcto uso del software, se recomienda lo siguiente:

Nivel básico (usuarios finales)

Para asegurar que puede disfrutar plenamente de sus derechos, el usuario final debería verificar:

- Que el paquete distribuido incluye una copia de la licencia GPL (y/o cualquier otra licencia de SFA incluida en el software).

- Que el paquete incluye una copia del código fuente del software, está a disposición en Internet o se proporcionará a petición del usuario.

Nivel avanzado (integradores, clientes finales sofisticados)

Por contrario, el usuario avanzado debería realizar un control más preciso del uso de SFA, sobre todo si se plantea la posibilidad de redistribuir el software, modificarlo y/o integrarlo en otras aplicaciones. Se recomienda verificar:

- Que el distribuidor es titular de todos los componentes de la aplicación o si incluye otros componentes bajo licencias de SFA.

93 La obligación de la GPL 2.0 de ofrecer el código fuente a cualquier tercero en el caso de no distribuir el programa con sus fuentes ha sido limitada en la GPLv3. Ahora, solamente es necesario ofrecer el código fuente al destinatario original del ejecutable.





- Que el paquete incluye el código fuente.
- Que la versión del código fuente es la que corresponde al binario.
- Que el paquete incluye una copia de la licencia del componente bajo la GPL.
- Que el paquete incluye los scripts y otros elementos necesarios para recompilar e instalar el software.

4.4.2. Garantías sobre SFA

Los sistemas informáticos son herramientas críticas en la actividad de una empresa, y la gestión de riesgos en términos de garantías y responsabilidades es un factor que se debe valorar la hora de elegir un programa o solución. Se han presentado varios argumentos contrarios al uso del SFA que presentamos y contestamos en la siguiente tabla:

ARGUMENTO	RESPUESTA
Las licencias de SFA incluyen pactos cuyo propósito es limitar las garantías de funcionamiento o conformidad del programa, evitar responsabilidades derivadas de infracciones de derechos de terceros y exonerar a los proveedores de responsabilidades en relación al uso del software.	Las licencias privativas (sobre software estándar) suelen incluir las mismas exclusiones que las licencias de fuentes abiertas. Para casos especialmente críticos existen empresas locales de soporte (o el mismo proyecto) que ofrecen garantías adicionales.
Los licenciantes de SFA (individuos, proyectos o comunidades) suelen tener pocos recursos para hacer frente a determinadas responsabilidades.	Hay intermediarios (tales como Red Hat o Novell para GNU/Linux, o empresas locales de soporte) que transforman esta supuesta debilidad en un modelo de negocio. Basan sus servicios (de pago) en la provisión de garantías en cuanto a la corrección de errores, soporte y evolución del programa. Son proveedores frente a los cuales el usuario podría exigir responsabilidades.
No hay ninguna garantía (en su acepción no jurídica) de que el proyecto que crea el software tenga futuro y continuidad o que se adapte a nuevas plataformas o tecnologías.	La organización usuaria tiene acceso al código fuente del programa que, junto con un derecho de copia y modificación, le da independencia frente al proveedor, tanto por lo que se refiere a la necesidad de aplicar correcciones[94] como a la evolución del software. Notemos que tampoco hay garantías de que una empresa proveedora de software privativo continúe mejorando o desarrollando el software (prestando mantenimiento o corrigiendo errores).

94 Un derecho que el marco jurídico impone para cualquier tipo de software.





CAPÍTULO 5

PROYECTOS DE CREACIÓN FOSS





Ficha resumen

OBJETIVO

Presentar los principales aspectos jurídicos que pueden surgir durante el desarrollo y la gestión de un proyecto de creación de SFA.

DESTINATARIOS

Los creadores y desarrolladores de software que crean programas con la intención de liberarlos (tanto en el sector privado como en la Administración Pública).

RESUMEN

Los titulares de software pueden plantear su liberación bajo una o más licencias de fuentes abiertas.

Al integrar componentes de SFA de terceros, los proyectos pueden disfrutar de todos los derechos otorgados por las correspondientes licencias, pero deben tener en cuenta la compatibilidad de las obligaciones asociadas en caso de redistribuirlo.

Los principales aspectos jurídicos relevantes para un proyecto SFA son:

- La titularidad de derechos del software del proyecto.
- La compatibilidad de licencias del SFA incorporado en el proyecto, entre ellas y con la del proyecto.
- Las obligaciones de licencias SFA a la hora de redistribuir el software.
- La selección de la licencia de distribución.
- La preparación del código: cabeceras, textos legales.
- La protección y política de marca.
- Las políticas y acuerdos sobre contribuciones.

REFERENCIAS

En la Guía:

- Capítulos 2 y 3 sobre licencias SFA y selección de licencia.
- Capítulo 7 sobre derechos de autor.
- Capítulo 8 sobre marcas.

En la red:

- Historia de Netscape/Mozilla en Voices. (*http://oreilly.com/catalog/opensources/book/toc.html*)
- Proyecto Openbravo: *www.openbravo.com*
- Proyecto Campus: *www.campus.cat*





5.1. PROYECTOS DE SOFTWARE DE FUENTES ABIERTAS

La idea que el software de fuentes abiertas se desarrolla por grupos de hackers o desarrolladores independientes, de manera colaborativa, pero desorganizada, es incorrecta. En la gran mayoría de los casos, el SFA se desarrolla en el marco de un proyecto organizado, con un líder y una hoja de ruta, roles, responsabilidades y tareas asignadas (o asumidas) de manera coordinada. Todo soportado por una infraestructura tecnológica para la comunicación entre los miembros del equipo y con la comunidad, para la gestión del repositorio y de las versiones del código, y su distribución al público. Por lo tanto, hay un alto nivel de gestión y coordinación.

En esta Guía, cuando hablamos de un proyecto de SFA no nos referimos solamente a un "proyecto" en el sentido específico de un medio de organización de varias personas o entidades que colaboran para desarrollar un software de fuentes abiertas (como, por ejemplo, los proyectos Apache, Mozilla, KDE, GNU o GNOME[95]). Utilizamos este concepto en un sentido más amplio, para incluir cualquier proyecto personal, comercial, universitario o del sector público, el resultado del cual sea la creación de un programa que va a distribuirse bajo una licencia de SFA.

Proyectos de SFA

Un proyecto puede ser llevado a cabo por una sola persona (por ejemplo, un estudiante con un proyecto de fin de carrera); un grupo de personas (un grupo de investigación o de compañeros de un LUG[96]); una empresa (Red Hat Inc., Canonical Ltd.[97]); un consorcio de entidades privadas y/o públicas (Campus, ObjectWeb[98]) o una administración pública (Linex y Linkat, entre otros[99]).

El propósito de este Capítulo es plantear aquellos aspectos jurídicos que intervienen en la gestión de un proyecto de SFA. Presentaremos tres casos, con crecientes niveles de complejidad:

• **Proyecto básico:** se libera un software de nueva creación.

• **Proyecto complejo:** se integra componentes de SFA de terceros.

• **Proyecto maduro:** se ha publicado el código y la comunidad empieza a contribuir. Estos proyectos deberían tener cuatro objetivos jurídicos principales:

95 En Internet, respectivamente, en www.apache.org, www.mozilla.org, www.kde.org, www.gnu.org/home.es.html, www.es.gnome.org.
96 Siglas en inglés de Linux User Group (grupo de usuarios de Linux). Hay ejemplos en http://www.linux.org/groups/.
97 En www.redhat.com y www.canonical.com.
98 En http://www.lafarga.org/campus/cat/index.html y http://www.objectweb.org/.
99 En www.linex.org/ y http://linkat.xtec.net/portal/





1. Asegurar a **titularidad de los derechos** necesarios y suficientes para poder liberar el software bajo licencia SFA.

2. Seleccionar una **licencia de distribución** que asegure los objetivos del proyecto.

3. Cumplir con las **obligaciones establecidas** en las licencias sobre componentes de SFA integrados en el software y evitar su incompatibilidad.

4. Proteger la **reputación** del proyecto y la calidad de su software.

Relevancia de los aspectos jurídicos en un proyecto de SFA

Aunque en la presente guía se identifican una serie de aspectos jurídicos a tener en cuenta, no todos los aquí mencionados serán relevantes para cada proyecto de SFA, ni tendrán un impacto necesariamente determinante para el mismo. Sin embargo, y a pesar de que no ha habido ningún caso ante los tribunales españoles sobre estos temas (con respecto al software de fuentes abiertas), consideramos que su descuido puede influir de manera importante en el éxito del proyecto en cuestión.

5.2. CASO 1: EL PROYECTO BÁSICO

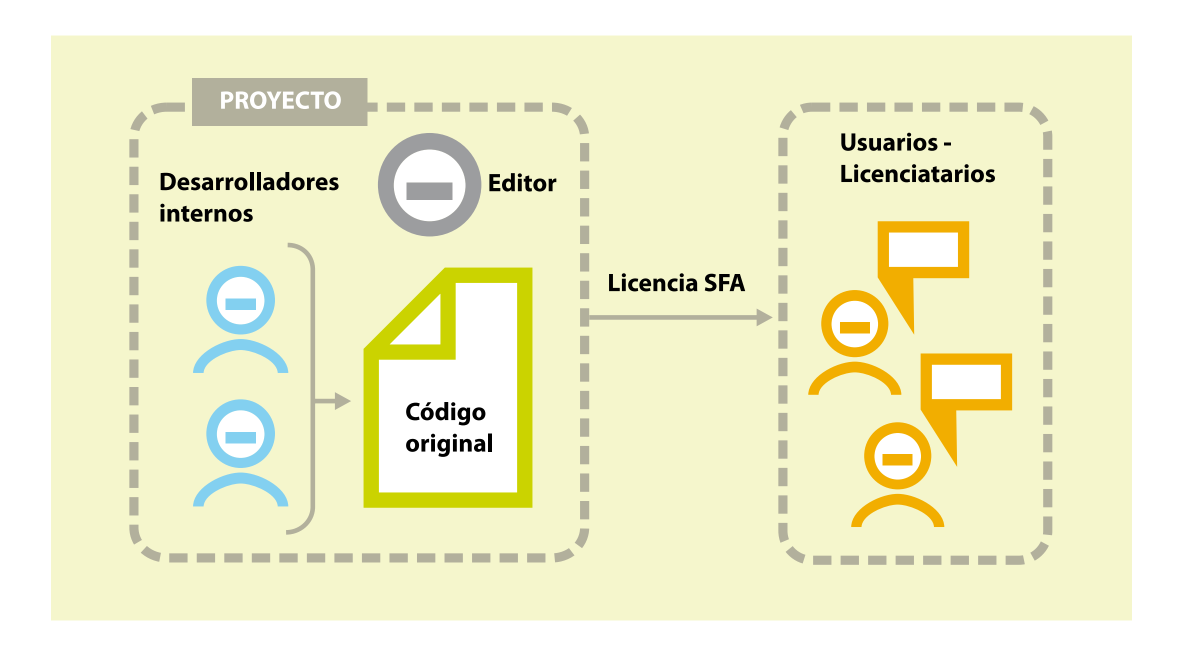
En un proyecto básico el software es de nueva creación y no integra ningún componente de terceros. Este caso es singular, en el sentido de que hoy en día casi todos los proyectos se basan o integran componentes de SFA de terceros (librería, rutinas, scripts) con el objetivo de no volver a crear lo que ya se ha creado. Aún así nos sirve para comenar algunos aspectos jurídicos básicos.

En este gráfico, el "editor" es la persona responsable del desarrollo y divulgación del software: puede ser una sola persona, un grupo de personas, una empresa o una entidad pública (ver más adelante). Los desarrolladores internos son empleados de la entidad (o miembros del equipo del editor). El software se desarrolla internamente (es decir, dentro de la empresa o entidad o entre los miembros del grupo de desarrollo) y luego se publica bajo licencia de SFA.





Esquema habitual de desarrollo y distribución de FOSS en un proyecto simple



Principales aspectos jurídicos de un proyecto básico	
En el desarrollo:	**En la liberación / distribución:**
• Garantizar la titularidad original del software.	• Selección de la licencia.
• Evitar los riesgos de patentes.	• Selección y registro de una marca.
• Delimitar el perímetro de confidencialidad.	• Distribución en línea.

5.2.1. El desarrollo

Para distribuir legalmente un programa o cualquier otra obra (como la documentación técnica o los manuales) bajo licencia SFA, el editor[100] debe asegurarse de que tiene los derechos de propiedad intelectual suficientes para hacerlo. Para ello, debe ser titular originario de los derechos del software (como autor o titular originario por presunción legal) o licenciante de los derechos del software creado por terceros (proveedores, subcontratas, becarios, etc.) que deberán permitir, a su vez, la posterior distribución del programa bajo la licencia del proyecto.

100 Para comentar los proyectos de creación de SFA desde una perspectiva jurídica debemos introducir el concepto de editor del software, haciendo referencia a la persona (física o jurídica) que coordina la creación y publicación del programa, es decir, que gestiona su desarrollo y realiza su distribución y comunicación pública. Puede ser el desarrollador o grupo de desarrolladores que cuelga el código en su propia página web, la universidad que publica los resultados de una investigación llevada a cabo por su personal o la empresa que distribuye un software desarrollado por sus empleados.





Derechos de autor

El desarrollador independiente que no usa software de terceros (es decir, que lo crea desde cero) será titular de los derechos de autor del programa. Lo mismo ocurre en un proyecto empresarial en el cual los empleados de la empresa desarrollan todo el código del programa. Los derechos sobre este software creado internamente, es decir por un autor, un grupo de coautores, o por un equipo de empleados dentro de una empresa, serán del autor, el grupo de autores o el empresario (respectivamente).

Derechos del titular

Siendo titular de los derechos, el editor podrá, entre otras cosas, seleccionar libremente la licencia de SFA bajo la cual distribuir el software, proceder a su distribución (en Internet, a un cliente) y defenderlo contra infracciones.

También puede ser que participen en el desarrollo personas ajenas al proyecto: consultores externos o subcontratas, proveedores, becarios universitarios, etc. En estos casos, en ausencia de una cesión expresa y por escrito, la titularidad de los derechos del trabajo de terceros será de estas personas y no de las entidades donde trabajan, proporcionan servicios o estudian. Por lo tanto, el editor deberá asegurar la cesión de los derechos al proyecto (normalmente por contrato o convenio) suficientes para poder liberar el software bajo la licencia de SFA.

Titularidad de proyectos empresariales / institucionales

Los proyectos empresariales e institucionales (y los universitarios en particular) deben cuidar especialemente la cuestión de la titularidad del software creado por distintas personas de su entorno. El software creado por el personal fijo de la institución (empleados o funcionarios, como profesores o investigadores), salvo pacto contrario en su contrato laboral, serán propiedad de la entidad. Si el autor no es empleado de la entidad, tendrá la titularidad originaria de sus creaciones y los derechos deberán ser cedidos a la entidad para que ésta pueda explotar el software.

Sin embargo, puede haber casos en los cuales las personas que participan en el desarrollo no sean titulares originarios de los derechos de sus aportaciones. Por ejemplo, cuando:

Un desarrollador del proyecto ha “cortado y pegado” código desde Internet;



el código aportado es realmente un software de un tercero bajo licencia (privativa o de SFA); o un desarrollador externo aporta código a un proyecto siendo empleado de otra entidad, habiendo desarrollado su tarea dentro de su propio marco laboral, y sin formalización de la cesión por parte de su empleador.

Estas situaciones, que surgen igualmente en el caso de desarrollos privativos, se resuelven sencillamente con una adecuada gestión (y cesión de derechos) de las aportaciones.

Snippets, parches, etc.

Existen muchos lugares en Internet en los cuales se publica y comparte código (snippets, scripts, parches, etc.). Es muy común (y eficiente) aprovechar este código par aportar funcionalidades o resolver problemas sin tener que volver a desarrollarlo desde cero. El proyecto deberá considerar este software "de fuente externa" y asegurarse de que tiene los derechos suficientes para utilizarlo dentro del proyecto (por acuerdo de cesión de derechos o por licencia SFA).

Propiedad industrial y confidencialidad

Los aspectos jurídicos del proyecto no conciernen únicamente a los derechos de autor, sino también a diferentes figuras de la propiedad industrial y la protección de los secretos industriales:[101]

1. En el momento del desarrollo, en el caso de inventar un nuevo proceso o producto susceptible de ser patentado, el proyecto podría pensar en solicitar una patente sobre un proceso inventivo (en el caso de que se considere útil, ético o hasta necesario solicitar patentes sobre software, algo dudoso cuando hablamos de SFA). Este aspecto está comentado con más detalle en el Capítulo 8 de esta Guía, pero en general una condición esencial es mantener la confidencialidad de la invención hasta la solicitud de patente.

2. Asimismo, es posible que determinada información confidencial generada durante el desarrollo del programa tenga cierto valor comercial, es decir, que su explotación genere ingresos para sostener el proyecto (para los desarrolladores, o para una empresa que se base en este software). El titular de esta información, que suele ser el editor o la empresa en la cual se desarrolla el software, deberá cuidarse de mantener su secreto, a través de medidas tecnológicas (cifrado, controles de acceso) o legales (como las obligaciones de confidencialidad que sujetan a los empleados o los acuerdos de confidencialidad con terceros).

101 *Ver el capítulo 8 de esta Guía para un resumen de estos aspectos.*





5.2.2. La liberación

Un vez desarrollado el software (o durante un proceso de desarrollo colaborativo), existirá una voluntad de liberarlo, es decir, distribuirlo bajo una licencia de SFA. Para ello, será necesario buscar un nombre, seleccionar una licencia de fuentes abiertas para el programa y establecer la plataforma de distribución.

• En cuanto a la protección jurídica del nombre del software, el proyecto podrá registrar un signo (denominativo o gráfico) como **marca**, para identificar y distinguir el software y su origen.[102]

Iconografía

Los proyectos de SFA suelen identificarse con iconos muy reconocidos hoy en día, pero no todos serán marcas registradas (igualmente están protegidos por derechos de autor, por ser obras gráficas).



La marca protegerá la reputación (calidad) del software, identificará su origen y, a la vez, generará valor, aumentando el interés económico de un proyecto comercial de SFA. La marca sirve, por ejemplo, para impedir que terceros usen un signo para redistribuir versiones modificadas del software de inferior calidad o en direcciones de Internet para promocionar su propio negocio.

Distribución / comunicación del software a nivel mundial

Se debe evitar nombrar el proyecto con una marca registrada por un tercero, por lo menos en el país original del proyecto y, eventualmente, en cualquier parte del mundo. En una ocasión, la distribución internacional de un SFA en Sourceforge permitió a un tercero, titular de una marca similar al nombre del SFA, alegar que se estaba usando indebidamente su marca en los países donde la tenía registrada. Por ello exigía el cese de su uso, petición que el proyecto tuvo que acatar. Pocos proyectos tendrán los fondos para defenderse. Tampoco querrán limitar la distribución del software en los países protegidos. En este caso, a menudo, la solución más rápida es cambiar el nombre del proyecto.

102 *Ver el Capítulo 8 de esta Guía, sobre Marcas.*





• La **selección de licencia** merece un apartado por si solo, por lo que le hemos dedicado el apartado 2 del Capítulo 3.

• Finalmente, el software será **distribuido en Internet** y, por lo tanto, su publicación esatará sujeta, como cualquier plataforma web, a las obligaciones legales que rigen los sitios y las actividades en Internet, y la LSSICE[103] en particular. Esta ley impone obligaciones básicas en cuanto a la identificación del titular de la plataforma y cualquier proceso eventual de contratación. Se recomienda, en particular, establecer un aviso legal (incluyendo aspectos relacionados con los enlaces, la limitación de responsabilidades y la protección de datos de carácter personal, si se recoge cualquier dato de esta naturaleza) y es buena práctica indicar la licencia SFA del software y las reglas de uso de la marca.

5.3. CASO 2: UN PROYECTO QUE USA COMPONENTES DE SFA

Este segundo caso es más habitual y los aspectos jurídicos son especialmente importantes. Se trata de un proyecto que integra componentes de SFA de terceros, ya sea para ampliar, integrar o mejorar las funcionalidades de un programa existente (como el proyecto GONG, que comentamos a continuación) o para incorporar funcionalidades a un software propio. Por ejemplo, unas librerías de software, una base de datos, o un conjunto completo de subcomponentes, incluyendo un sistema operativo, una base de datos y un servidor de aplicaciones o web[104].

Software compuesto

Mirando, por ejemplo, los componentes de Zimbra (www.zimbra.com), vemos que integra todo un conjunto de subcomponentes de SFA bajo diferentes licencias, listados en http://www.zimbra.com/license/zimbra_third_party_licenses_2.1.html.

Otros proyectos, como Sakai, publican un listado de licencias usadas, compatibles y permitidas. (http://bugs.sakaiproject.org/confluence/display/LIC/3rd+Party+Licenses+and+Software).

El siguiente gráfico ilustra el rol de las personas implicadas en un proyecto complejo y sus relaciones legales. La correspondiente tabla resume los principales temas jurídicos adicionales a los identificados en el proyecto básico.

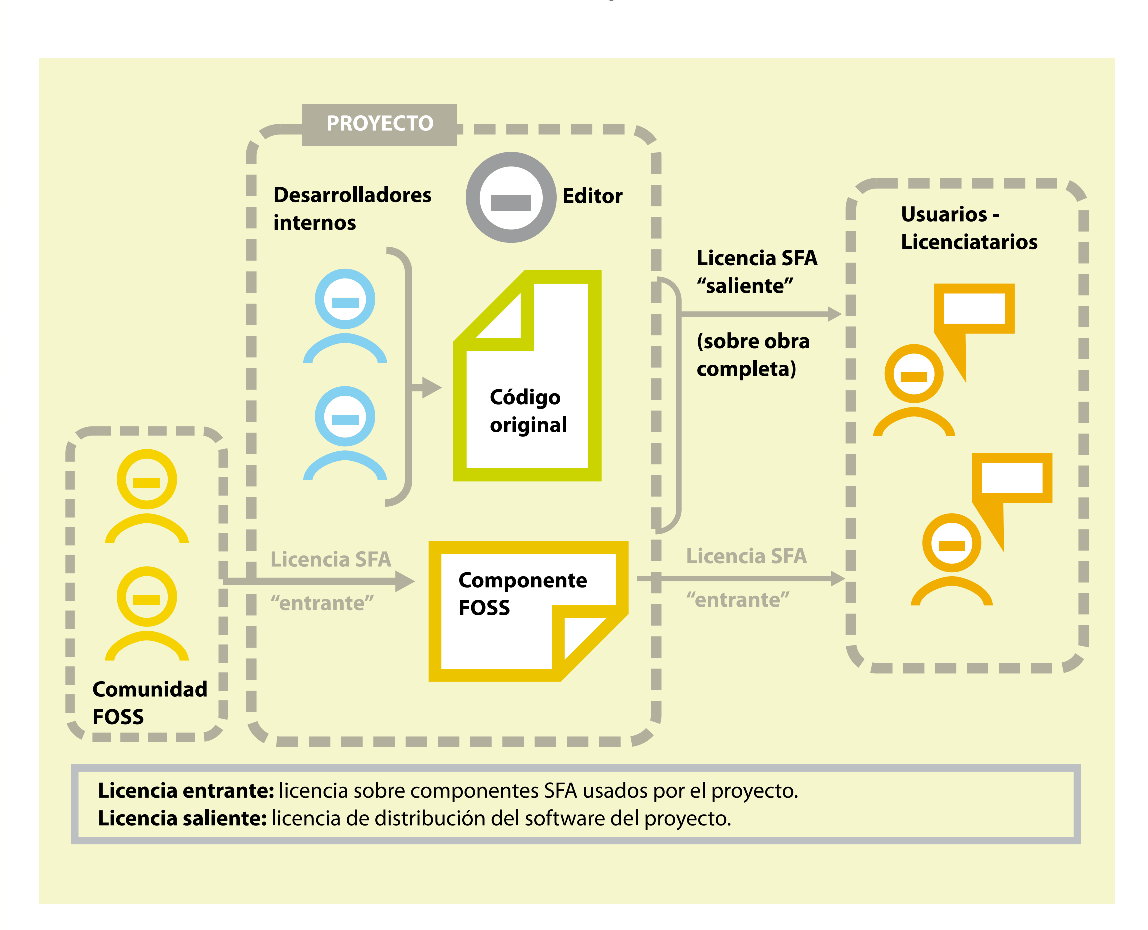
103 Ley 34/2002, de 11 de julio, de servicios de la sociedad de la información y de comercio electrónico.
104 Como el "stack" llamado LAMP constituido por GNU/Linux, Apache, MySQL y PHP o Perl.





En el gráfico identificamos un elemento adicional respecto al primer caso: el uso de componentes de SFA creados fuera del proyecto (por la comunidad FOSS) y licenciados como parte del mismo. Cada componente tiene una licencia originaria –la licencia entrante - que deberá ser compatible con la licencia del proyecto (saliente), tal y como comentamos en el capítulo 3.

Esquema habitual de desarrollo y distribución de FOSS con usos de componentes FOSS



Principales aspectos jurídicos de un proyecto básico

Desarrollo

• Licencia entrante sobre SFA:

- Identificación de la misma.

- Determinar su compatibilidad con la licencia de saliente.

• Existen patentes sobre componentes de software?

Distribución

• Cumplimiento de las condiciones impuestas por las licencias sobre componentes.

• Uso de marcas de componentes en la documentación / publicidad del proyecto.





5.3.1. El uso de SFA de terceros

Como vemos en la tabla, los nuevos aspectos jurídicos que un proyecto complejo debe tener en cuenta están relacionados con el uso de los componentes de SFA disponibles en Internet. Pensamos, por ejemplo, en los ficheros del OpenJDK de Sun Microsystems,[105] los paquetes y programas de proyectos como Apache o Eclipse, o cualquier otro programa disponible en repositorios de SFA como Sourceforge o Freshmeat.[106]

La regla básica es que el proyecto deberá obtener los derechos suficientes para poder (1) integrar el componente de SFA en el proyecto y luego (2) distribuirlo a terceros. Estos derechos se derivan de las licencias sobre componentes utilizados:

- **Las licencias SFA siempre conceden los derechos suficientes para crear el software del proyecto:** conceden los derechos para copiar y modificar el software,[107] que son los necesarios para el desarrollo y uso interno del programa[108].

También proporcionan los derechos de distribución y comunicación pública y, por lo tanto, permitirán la redistribución del componente, junto con o como parte del nuevo programa, pero sujeta a condiciones que varían según la licencia[109].

El principal requisito para redistribuir el componente como parte del proyecto **es que las obligaciones dispuestas en la licencia (entrante) sobre el componente y en la licencia (saliente) del proyecto sean compatibles.**

Esto es todavía más importante cuando se usan componentes bajo licencias con copyleft, o bajo la GPL en particular. Esta licencia exige que cualquier modificación del software y cualquier programa que lo integre (dependiendo del modo de integración), se distribuya bajo esta misma licencia. Por lo tanto, la mayoría de los proyectos que aprovechan componentes licenciados bajo la GPL adoptarán esta misma licencia (o excepcionalmente la Affero GPLv3, si se trata de la GPLv3).

105 *En http://openjdk.java.net/*

106 *En www.sourceforge.net y www.freshmeat.net. Hay muchos otros repositorios o forjas, como www.lafarga.cat, www.osor.eu. Pero el FOSS es omnipresente en Internet y muchos proyectos tienen su propia página web: Jboss, Ruby, PHP, Python, Plone, Joomla!, etc.*

107 *Los derechos de "reproducción" y "transformación", según la LPI.*

108 *Incluso, en este sentido, se puede integrar SFA bajo licencias incompatibles, pero el resultado no podrá distribuirse.*

109 *Tal y como lo hemos comentado en el Capítulo 2 sobre los diferentes tipos de licencia.*





Compatibilidad legal de componentes de SFA

Nos remitimos al Capítulo 3 de esta Guía, que versa sobre la compatibilidad entre componentes. En la práctica, una licencia de fuentes abiertas permisiva permitirá integrar un componente en un programa distribuido prácticamente bajo cualquier licencia, de SFA o no libre. Por contra, un componente licenciado con copyleft fuerte (como la GPL) solamente podrá distribuirse integrado en un programa bajo la misma licencia.

El software GONG[110], por ejemplo, se ha creado sobre la base de Alfresco ECMS, que se distribuye bajo la GPL, pero con una excepción especial[111]. Así, cumpliendo con las obligaciones del efecto copyleft, la licencia de la distribución de GONG es la misma GPL, incluyendo esta excepción.

Para evitar problemas en este sentido (cualquier conflicto entre licencias impedirá la redistribución del componente como parte del proyecto) existen una serie de buenas prácticas que, llevadas a cabo, mejorarán la gestión de los componentes de SFA. En particular, se recomienda que los miembros del proyecto identifiquen las licencias de los componentes que se van a usar y verifiquen su compatibilidad entre ellas y con la del proyecto, en función de cómo se vayan a integrar en el mismo.

110 En http://gong.morfeo-project.org/lng/es
111 http://wiki.alfresco.com/wiki/Main_Page





Otras cuestiones relativas a la selección y uso de componentes de SFA

A continuación exponemos la lista de preguntas que los responsables de un proyecto deberían considerar a la hora de integrar o incluir un componente SFA. Estas cuestiones no sólo están relacionadas con la operativa del propio proyecto sinó también con la estrategia de negocio de la empresa que lo financia:

A nivel práctico:

- ¿Se han leído y entendido los términos de la licencia y guardado una copia de la misma?
- ¿Se va a transformar (crear una "obra derivada" de) el componente, o se incluirá meramente como librería? ¿Qué implicará una u otra opción a la hora de distribuir el resultado?
- ¿En caso de transformar el componente, las condiciones sobre la redistribución de obras derivadas son compatibles con la licencia del proyecto?
- ¿El licenciante proporciona el código fuente? ¿Se necesita para el uso del programa y/o a la hora de distribuirlo?
- ¿Es realmente necesario distribuir el componente junto con el programa? ¿O es un componente que los usuarios pueden descargar por separado, como parte de la plataforma de instalación o ejecución?
- ¿La licencia SFA incluye una licencia de patente?

A nivel estratégico:

- ¿Se contempla dar una garantía (a los usuarios finales) sobre el software del proyecto más amplia que la garantía que se recibe sobre este componente? ¿El licenciante ofrece un plan de garantías adicionales (soporte de segundo nivel, etc.)?
- ¿El proyecto o el titular del componente publica criterios de calidad técnica (o tiene una reputación correspondiente) en relación con el desarrollo del componente? ¿Tiene una hoja de ruta, u otra indicación de continuidad en el tiempo, para el mantenimiento y evolución del producto?
- ¿El proyecto / titular del componente demuestra algún indicio de calidad legal (diligencia a la hora de cuidar los aspectos jurídicos comentados en esta Guía) o es un snippet cortado de un foro de Internet?
- ¿Cuáles serían los riesgos e impactos para el proyecto si finalmente no se pudiera usar y redistribuir este componente (por razones relacionadas con los derechos de autor o de patente), si tuviera errores de funcionamiento o si los responsables del componente lo abandonaran y no proporcionara más soporte sobre el mismo?





Finalmente, también se debe evaluar el riesgo de que un tercero haya patentado un proceso implementado en un componente que se quiere usar (sobre todo en un contexto en el que las patentes podrían prevalecer). El titular de una patente de software, siempre que ésta sea válida, podrá impedir la distribución y explotación de un programa que implemente el proceso en cuestión.[112]

5.3.2. La redistribución de componentes de SFA

A la hora de distribuir un programa que incluye componentes SFA de terceros surgen dos aspectos jurídicos que es importante tener en cuenta (más allá de las condiciones legales relativas a los derechos de propiedad intelectual comentados en el apartado anterior):

1. Obligaciones sobre la redistribución

Por un lado, las licencias de SFA suelen incorporar una serie de condiciones prácticas adicionales que el proyecto deberá respetar a la hora de distribuir el programa final. Estas condiciones pueden incluir, entre otras, la obligación de:

• mantener el aviso de los derechos originales sobre cualquier componente (copyright notice) y las cláusulas de exención de responsabilidades (disclaimers),

• incluir una copia de la licencia del componente en la distribución,
indicar el alcance y/o la fecha de las modificaciones realizadas,

• respetar la obligación o prohibición de uso de una marca u otra forma de atribución en la documentación, y

•en algunos casos y de manera excepcional, reportar al titular original las modificaciones (por ejemplo, el Artículo 4.a de la licencia *Artistic License 2.0*).

Gestión de la distribución

Será útil establecer un listado (del tipo checklist) de estas obligaciones a partir de las licencias originarias, cuyo cumplimiento se verificará antes de publicar el programa. Este listado irá creciendo a medida que el proyecto use más componentes de terceros. En un proyecto grande, con cientos de componentes, esta tarea puede suponer un esfuerzo significativo y requerir la asistencia de abogados especialistas en el tema.

112 Ver el apartado sobre patentes de software del Capítulo 8 de esta Guía.





2. Derechos de marca

Por otro lado, ya hemos comentado que el Derecho de marcas interfiere también en la gestión de un proyecto SFA, en particular en la protección del signo usado para identificar el software. También se deberá prever el uso de marcas de terceros, ya sea en la misma interfaz gráfica del software, en sus actividades publicitarias o en la documentación del producto.

Uso de marcas de terceros

Es frecuente que se indique que un software se basa en, usa o es compatible con un programa en particular (una base de datos, un sistema operativo) o utiliza un tecnología específica. Por tanto, los proyectos tienden a hacer referencia a nombres de producto o empresa protegidos por derechos de marca, como Java®, Jini®, Sun®, Oracle®, JBoss®, MySQL®, Windows®, el logotipo Open Source, etc., ya sea en la interfaz del usuario del software o en su documentación.

Para utilizar una marca de un tercero, en la promoción o distribución del software del proyecto, será necesario obtener una licencia por parte del titular. Para facilitar este trámite, muchas empresas y proyectos de SFA (como MySQL, SugarCRM, Alfresco, Debian, GNOME, KDE, RedHat, etc.) publican "condiciones de uso establecidas" en una política de marca expresa[113].

113 Por ejemplo: MySQL: http://www.mysql.com/about/legal/trademark.html, o Sun Microsystems Inc. en relación con Java: http://www.sun.com/policies/trademarks/

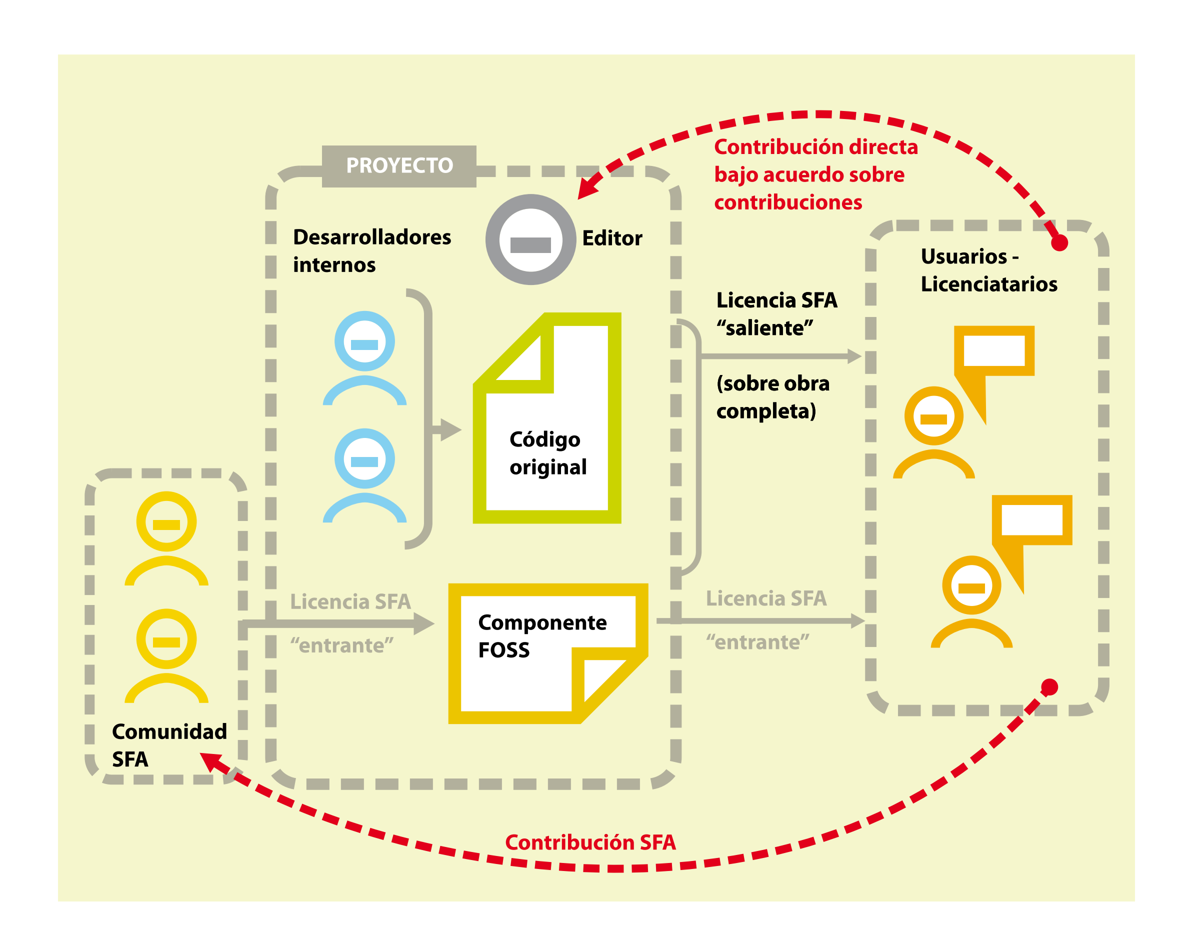




5.4. CASO 3: UN PROYECTO MADURO CON CONTRIBUCIONES

El tercer caso se refiere a un proyecto ya liberado que recibe contribuciones de la comunidad. Explicamos las relaciones que se establecen entorno a un proyecto maduro en el siguiente gráfico, junto con la tabla resumen de aspectos jurídicos adicionales que se deben considerar:

Esquema habitual de desarrollo, distribución y contribución de SFA







Destacamos, en rojo, un elemento adicional repecto a los que hemos visto anteriormente: el proceso de contribución de software al proyecto por la vía directa (acuerdo sobre contribuciones) o indirecta (como componente de SFA de terceros).

Aspectos jurídicos adicionales en el caso de recibir contribuciones

Titularidad y cesión de derechos de las contribuciones al proyecto:

1. Averiguación del origen.

2. Cesión completa o licencia de uso bajo la licencia del proyecto / licencia permisiva.

3. Acuerdo sobre contribuciones.

En este caso se deberán gestionar correctamente las contribuciones de los miembros de la comunidad, personas ajenas al proyecto, para que éste tenga los derechos suficientes para su redistribución. Normalmente, existen dos formas de aportar código y garantizar los derechos necesarios del proyecto:

1. Realizar la aportación bajo la licencia del proyecto o bajo una licencia de SFA compatible, más permisiva (será el caso, a menudo, para aportaciones de la comunidad).

2. Ceder al proyecto los derechos sobre el software (mediante un acuerdo de cesión) con condiciones compatibles su licencia. Por lo menos, se deberán ceder los derechos de reproducción, transformación, comunicación pública y distribución de la aportación, sin limitaciones.

Acuerdos de cesión de derechos sobre contribuciones

En algunos proyectos de SFA, en particular en aquellos de carácter empresarial que también podrían distribuir el software bajo una licencia no libre (como MySQL o Alfresco), se exige que la firma de un "*Acuerdo sobre Contribuciones*" entre las partes. En él se establecerá la cesión irrevocable, exclusiva o no exclusiva de todos los derechos de autor (y, si es posible, de patente) o un régimen de cotitularidad. Además, suelen incluir garantías en cuanto al origen de la aportación y obligaciones de colaboración en caso de tener que proteger el software.

Ejemplos de acuerdos sobre contribuciones son el "ICLA" de Apache (*www.apache.org/licenses/icla.txt*) y el acuerdo sobre contribuciones para OpenOffice.org (en *www.sun.com/software/opensource/contributor_agreement.jsp*).





Los acuerdos sobre contribuciones tienen la ventaja de dar al proyecto la titularidad exclusiva de los derechos. Esto permite hacer cambios de licencia de SFA (de la GPLv2 a la GPLv3, por ejemplo) sin tener que solicitar autorización a todos los contribuidores, distribuir el software en régimen de licencia dual[114] y perseguir a los infractores que explotan el software violando su licencia de distribución (la legitimación activa).

5.5. LA LIBERACIÓN DEL SOFTWARE: UN CHECKLIST

El proyecto estará en condiciones de liberar el producto una vez:

• Asegurados los derechos del software,

• seleccionada la licencia de SFA para su distribución (respetando las obligaciones de las licencias sobre componentes),

• y, en su caso, elegido un nombre único y original.

Es cierto que el cumplimiento de las condiciones legales no es suficiente para asegurar el éxito de un proyecto de SFA (para ello hay que tener en cuenta otros aspectos técnicos, comunitarios, sociales y hasta comerciales), pero es necesario.

Antes de liberar el código es recomendable preparar su distribución desde la perspectiva jurídica. Para ello hay que asegurarse de que el programa a distribuir tiene las indicaciones necesarias para proteger los derechos de los titulares (el proyecto/editor) y de que cumple con todas las obligaciones relevantes de las licencias de sus componentes.

Revisión legal:

A la hora de publicar el software, se recomienda llevar a cabo, como mínimo, las siguientes acciones:

1. Revisar la lista de componentes y sus correspondientes licencias y obligaciones sobre la distribución.

2. Incluir las cabeceras de los ficheros de código original, y los avisos de autoría y de licencia del proyecto.

3. Mantener intactas las cabeceras de los ficheros de SFA que no han sido modificados.

4. Agregar una anotación en las cabeceras de los ficheros de SFA que hayan sido modificados (agregando la naturaleza, fecha, autor y titular de los derechos de la modificación y un contacto).

5. Redactar un fichero tipo "licensing.txt" con la información jurídica sobre el producto (componentes, licencias del proyecto y de los componentes, avisos obligatorios, e-mail de contacto).

114 Ver el Capítulo 3 de esta Guía.





6. Crear una carpeta "legal" para poner la documentación jurídica ("notice.txt", "licensing.txt", licencia del producto y las licencias SFA de los diferentes componentes).

7. Calcular y publicar el valor *hash*[115] de la liberación para asegurar la integridad del software distribuido.

8. Asegurar que cualquier distribución en binario vaya acompañada del correspondiente código fuente (o una indicación sobre dónde encontrarlo).

Asimismo, exponemos las mejores prácticas a tener en cuenta:

Otros aspectos de la liberación

Es recomendable que el proyecto lleve a cabo las siguientes acciones:

1. Asegurar el reconocimiento de los autores en la sección "About" del programa (si la hay).

2. Incluir un proceso de aceptación de licencia (en el caso de que se trate de un ejecutable de usuario final) durante la instalación.

3. Redactar una política de uso del nombre y, en su caso, cualquier marca del proyecto.

4. Solicitar el registro de la marca del proyecto en las jurisdicciones relevantes (España[116], Europa[117], etc.).

5. Establecer una sección en la web del proyecto con la información legal útil para los usuarios: la licencia del producto, la lista los componentes de terceros incluidos en el mismo, una política de uso de marca, acuerdo sobre contribuciones, etc.

6. Determinar la licencia para los contenidos de la web y en particular su documentación (wiki, etc.).

7. Averiguar si el sitio web cumple con la ley vigente (la "LSSICE" y la "LOPD"[118] en particular) y, si los potenciales usuarios son consumidores finales, respetar las leyes sobre la protección del consumidor y usuarios[119].

5.6. EJEMPLOS DE PROYECTOS SFA

115 El hash es el resultado de una función algorítmica (normalmente, se utilizan las conocidas como MD5, SHA1, etc.) que se utiliza también en los sistemas de firma electrónica y que sirve para identificar, de una manera unívoca, un contenido por mediación del resumen o hash.

116 Oficina Española de Patentes y Marcas, en http://www.oepm.es

117 Oficina Española de Patentes y Marcas, en http://oami.europa.eu/

118 Respectivamente, Ley 34/2002, de 11 de julio, de servicios de la sociedad de la información y de comercio electrónico y Ley Orgánica 15/1999 de 13 de diciembre de Protección de Datos de Carácter Personal.

119 Real Decreto Legislativo 1/2007, de 16 de noviembre, por el que se aprueba el texto refundido de la Ley General para la Defensa de los Consumidores y Usuarios y otras leyes complementarias.



En este apartado, comentamos tres ejemplos de proyectos de creación y/o liberación de SFA, para ilustrar los aspectos jurídicos que puedan surgir y la importancia de su gestión.

5.7. COMENTARIOS FINALES

CASO	Nestcape / Mozilla
El proyecto	En 1998, Netscape decidió liberar el código de su navegador de Internet Navigator.
Cuestiones jurídicas	La liberación de Navigator planteó los siguientes problemas: **1.** Había más de 75 componentes de terceros en el software del navegador, con licencias específicas. **2.** El software se basaba en componentes Java, en su momento una tecnología privativa. **3.** Se requería un alto nivel de coordinación de los equipos trabajando en la preparación del código. **4.** El programa integraba software de cifrado (que no se podía liberar de acuerdo con las leyes americanas sobre la distribución de productos de cifrado). **5.** Se tenía que elegir una licencia adecuada para el proyecto, de acuerdo con los objetivos de la empresa: algo entre el *copyleft* de la GPL y la permisividad de la BSD. **6.** La empresa anticipaba la necesidad de gestionar la propiedad del código y las contribuciones de terceros después de su liberación.
Resolución	Para resolver estos problemas, Netscape emprendió las siguientes acciones: **1.** Se reunió con todos los titulares de los componentes de terceros para solicitar la autorización para liberar su código (en su defecto, era necesario eliminar o remplazar el componente). **2.** Se eliminaron todas las tecnologías Java. **3.** Desarrollaron un software de tipo *bugzilla* para la gestión y seguimiento del trabajo de los diversos equipos. **4.** Analizaron las licencias SFA existentes, para decidir sobre la redacción de una nueva licencia, la *Netscape Public License*, que reservaba algunos derechos específicos para Netscape. Esta licencia fue sometida a la aprobación de la comunidad, que la criticó. En consecuencia, se redactó la licencia *Mozilla Public License*, con *copyleft* suave. Al final, se reservó la licencia Netscape para los componentes del software originariamente liberados. Cualquier nueva contribución se realizaría bajo la MPL. **5.** Se creó la Fundación Mozilla, para el mantenimiento del código, la gestión el sitio web y la gestión de la comunidad Mozilla.





	Nestcape / Mozilla
Lecciones	El caso Netscape demuestra que es fundamental revisar el código fuente de una aplicación y las licencias sobre componentes de terceros para evitar cualquier infracción a la hora de la liberación. Asimismo, es importante coordinar los equipos que trabajan en la preparación del código para un correcto seguimiento legal de los trabajos (por ejemplo, para no olvidar ningún componente). En cuanto a licencia se refiere, vemos la importancia de elegir una licencia acorde no solamente con la estrategia de la empresa, sino también con la comunidad de desarrolladores e interesados. La consulta pública llevada a cabo a la hora de redactar la licencia nueva fue luego emulada por la FSF en la redacción de la GPLv3 (con mayor alcance).

CASO	Openbravo
El proyecto	Openbravo ERP es un software de gestión empresarial desarrollado por la empresa española Openbravo SL, de Pamplona. El software, creado desde el principio sobre la base de software de fuentes abiertas, es una aplicación web desarrollada siguiendo el modelo MVC (*Model, View, Control*) con tecnologías Java, y se distribuye principalmente en forma de código fuente.
Cuestiones jurídicas	El proyecto planteó una serie de cuestiones legales de las cuales destacamos las siguientes: **1.** El uso de componentes de SFA liberados bajo varias licencias libres, que podían resultar incompatibles entre sí. **2.** La creciente integración de componentes SFA, con una diversidad de licencias, para ofrecer funcionalidades adicionales. **3.** La selección de una licencia de distribución compatible con dichos componentes y acorde a la estrategia de negocio de la empresa. **4.** La preparación del código para su liberación (preparación de cabeceras, etc.). **5.** El desarrollo de ciertos componentes por expertos externos. **6.** La creciente participación de la comunidad, aportado parches y nuevos elementos de código. **7.** El registro y la protección de la marca “Openbravo” en diferentes jurisdicciones del mundo.





Openbravo

Resolución

El proyecto de publicación del software Openbravo ERP estableció tres etapas, para responder a estas cuestiones:

1. Verificación del código. Se revisó el code base de la aplicación para verificar la titularidad de la sociedad sobre el software y la compatibilidad de las licencias sobre los diferentes componentes de terceros integrados en la aplicación: partes de Compiere, varios componentes de Apache, Xinha, Dojo, CyberNeko HTML Parser, etc.

2. Selección de licencia y preparación de la distribución. Sobre la base de los componentes SFA integrados en el producto y de la estrategia "open source comercial" del proyecto, la empresa decidió usar una licencia de tipo MPL. Ésta es una licencia copyleft parcial o suave que permite a terceros crear productos ampliados basados en el ERP y distribuirlos bajo su propio régimen de licencia, mientras se mantenga el núcleo libre para la comunidad. La licencia añade la obligación de reconocer públicamente el origen del la aplicación en cualquier obra derivada. En base a esta licencia, se prepararon las cabeceras del código fuente y la documentación legal del proyecto para su publicación en Sourceforge.

3. Protección jurídica del negocio. Dado el impacto internacional del software, uno de los más descargados de Sourceforge y explotado en más de 48 países del mundo, ha sido importante registrar la marca "Openbravo" no solamente en España sino en Europa y otros países del mundo. Asimismo, la sociedad implementa una política de desarrollo para garantizar la calidad tanto técnica como legal del software.

Lecciones

El proyecto Openbravo nos demuestra la importancia de:

• Revisar de manera continua el código fuente para identificar los componentes SFA, determinar las obligaciones sobre su redistribución y, si fuera necesario, eliminar aquellos bajo licencias incompatibles o sin licencia identificable.

• Firmar acuerdos con desarrolladores externos y contribuidores al código.

• Elegir una licencia compatible con los componentes y acorde a la estrategia del negocio basado en el proyecto libre.

• Registrar la marca para defender la identidad del producto y proyecto, ofreciendo un sello de calidad.





CASO	Campus
El proyecto	El proyecto CAMPUS (*www.campus.cat*), promovido por la Generalitat de Cataluña, nace de la voluntad de ocho universidades catalanas de poder disponer de un campus virtual basado en software libre. Contempla las funcionalidades descritas en los estándares de LMS (*Learning Management System*). Tecnológicamente, el software se basa en un sistema central o infraestructura tecnológica básica y en un conjunto de módulos o servicios que se podrán desarrollar aparte (de forma independiente) del sistema central. La comunicación entre el sistema central y cada uno de los módulos se hará mediante llamadas basadas en las especificaciones OKI/OSID (*http://www.okiproject.org*), promoviendo la posibilidad de usar los sistemas Moodle o Sakai como sistema central.
Cuestiones jurídicas	El proyecto de desarrollo y publicación de CAMPUS planteó varios interrogantes: • La coordinación de la propiedad intelectual de más de 8 socios públicos y privados. • La centralización o no de la propiedad del código creado en el marco del proyecto. • La selección de la/s licencia/s de distribución del software. • La interrelación entre los componentes, el sistema central y el software de Sakai y Moodle, dos plataformas de e-learning ya publicadas con dos licencias originalmente incompatibles. Cómo implantar las mejores prácticas (legales) de desarrollo basado en un modelo libre de colaboración, en vez de un proyecto estructurado y jerárquico.
Gestión legal	Para hacer frente a estas cuestiones, el proyecto establece las siguientes pautas : • Un convenio flexible entre los socios que regula la propiedad intelectual: la propiedad no se centraliza en una institución, sino que cada socio es titular y responsable de su contribución, que se publicará a través del sitio web del proyecto. • Los componentes se entregan al coordinador (para su publicación) junto con una carta de compromiso en cuanto a la titularidad y compatibilidad legal de los componentes. • La licencia principal del proyecto es la GPL, pero se establece la posibilidad de incorporar componentes bajo otras licencias, de manera modular a través de un adaptador central (el OKI bus). Esto resuelve los problemas de integración de componentes bajo licencias incompatibles.





	Campus
Gestión Legal	La formación de los responsables de cada socio y la publicación de un manual para la gestión de aspectos legales. Una lista de discusión dedicada a temas legales y de comunidad, para responder a cualquier duda de los participantes del proyecto.
Lecciones	El proyecto CAMPUS, una colaboración de un gran número de instituciones, nos demuestra la importancia de tener relaciones flexibles entre los socios en cuanto a propiedad intelectual se refiere: no hace falta centralizar la propiedad en todos los casos, pero cada socio se responsabiliza de la legalidad de su contribución. Para ayudar a ello, el proyecto ofrece formación, directrices (jurídicas) de desarrollo, y asistencia en cuanto a la titularidad y el uso de componentes SFA. Es decir, establecer una gestión descentralizada de los aspectos jurídicos.





El objetivo de este Capítulo ha sido el de identificar los aspectos jurídicos que son relevantes para un proyecto de creación y/o liberación de SFA:

- Garantizar la titularidad de los derechos,
- seleccionar una licencia
- cumplir con las obligaciones de las licencias de los componentes
- y preparar el software para su distribución.

Los proyectos mejor gestionados (Apache, Mozilla, OpenOffice.org, Sakai, etc.) suelen establecer una serie de procedimientos y documentación que ayudan a garantizar la legalidad de su software, en cuanto a derechos de autor y el cumplimiento de las licencias sobre componentes. Entre otras cosas, establecen un acuerdo sobre contribuciones, una lista de licencias compatibles, un proceso de revisión de las licencias sobre componentes y las mejores prácticas a la hora de documentar las contribuciones.

La correcta gestión de los aspectos jurídicos debe estar al alcance de cualquier proyecto. Consideramos que esto es algo que se consigue,, con un buen diseño del procedimiento de creación, integración, modificación o migración del software, junto con un buen asesoramiento legal en el caso de que se considere necesario.





CAPÍTULO 6

INTEGRAR FSA





Ficha resumen

OBJETIVO

Presentar y comentar los aspectos jurídicos que afectan a la provisión de productos y servicios basados en SFA.

DESTINATARIOS

Distribuidores/integradores de soluciones y servicios basados en SFA.

Usuarios que adquieren sistemas y soluciones basados en SFA a través de un consultor o integrador de sistemas.

RESUMEN

Desde una perspectiva jurídica, el SFA ofrece beneficios significativos a los distribuidores/integradores de soluciones e, indirectamente, también a sus clientes:

El distribuidor/integrador disfruta de una amplia libertad a la hora de crear soluciones sofisticadas y adaptadas a las necesidades de sus clientes.Estos clientes tienen acceso al código fuente (con derechos de explotación), lo que les da una independencia frente a su proveedor y más importante, una garantía de estabilidad y continuidad tecnológica. Esta independencia obliga al integrador a ser más eficiente y competitivo.

Sin embargo, es necesario determinar correctamente los términos de distribución del software al cliente final. Para ello será imprescindible conocer las obligaciones que se pueden derivar de la integración de componentes bajo distintas licencias. Asimismo, el contrato de provisión del software debe adecuarse al uso del SFA, tanto por lo que se refiere a la propiedad intelectual como a los términos de acceso y mantenimiento del código fuente, garantías y responsabilidades.

REFERENCIAS

En la Guía:

- Capítulo 2, sobre las licencias de SFA.
- Capítulo 3, sobre la compatibilidad de licencias, garantías y responsabilidades.

Capítulo 4, sobre la perspectiva del cliente, como usuario final.

CASOS CITADOS

OpenTrends, Axiom Tech.





6.1. LA COMERCIALIZACIÓN DE SOFTWARE Y SERVICIOS BASADOS EN SFA

El SFA adquiere un papel cada vez más relevante a la hora de elegir software para crear una solución informática de tipo empresarial. Estos programas ofrecen prestaciones de calidad a todos los niveles, desde sistemas operativos, módulos de comunicaciones, bases de datos, y servidores de aplicaciones y web, hasta aplicaciones cliente sofisticadas como los sistemas de gestión empresarial y comercial (*ERPs y CRMs*), de gestión documental o de conocimientos (*DMS y KMS*), de Reporting y Business Intelligence (*BI*), de tramitación pública, de comercio electrónico, la gestión de contenidos web (*CMS*), etc.

El uso de este tipo de software ofrece al consultor-integrador varias ventajas relacionadas con el diseño, desarrollo e implementación de soluciones informáticas. No solamente suele ser de buena calidad[120] y gratuito, sino que confiere una gran libertad a la hora de integrar los diferentes módulos o componentes de una arquitectura más compleja y adaptarlo a las necesidades particulares del usuario final. Este es justamente el objetivo y filosofía que subyace tras el movimiento de software libre y de fuentes abiertas.
Este capítulo de la Guía se dirige especialmente a los consultores-integradores y presenta los principales aspectos jurídicos que afectan a la provisión de productos y servicios basados en SFA, tanto por lo que se refiere a la consultoría inicial como a los servicios de desarrollo, soporte o mantenimiento.

Modelos de empresas

Existen varias formas o modelos de prestación de servicios basados en SFA: proyectos originarios que prestan servicios de implantación y soporte sobre sus propios productos; empresas independientes que integran componentes de SFA para crear paquetes integrados (suites, sistemas de gestión, de seguridad, etc.); así como integradores que diseñan soluciones complejas a medida para sus clientes y/o prestan servicios de soporte, mantenimiento y formación.

Si bien cada una de estas modalidades tiene sus particularidades a nivel jurídico, nos limitaremos a presentar aquellos aspectos que son comunes a todas ellas.

6.2. PRINCIPALES ASPECTOS JURÍDICOS

120 Aunque no sea necesariamente el caso, en cuanto al grado de desarrollo (o madurez) del SFA, de los 178.000 proyectos actualmente disponibles en Sourceforge (diciembre del 2008), solamente 25.000 tienen un estatus de "5-Estable" y 500 son "6-Maduros" (igualmente, 25.500 son muchos!).





De forma muy similar a la provisión de servicios y aplicaciones privativas, el uso de SFA tiene varias implicaciones a nivel organizativo y legal, tanto para el proveedor como para el cliente. Las resumimos en la siguiente tabla:

6.2.1. Beneficios fundamentales

Implicaciones para el proveedor	Implicaciones para el cliente
Modelo de negocio • Venta de servicios de integración y desarrollo (no de licencias). • Servicios adicionales de soporte, mantenimiento y formación. • Aplicación o extensión de garantías sobre el software. • Normalmente, proyectos medianos y grandes. **Modelo de gestión del desarrollo** • Modular, con reutilización frecuente de componentes. • Flexible, abierto a la comunidad (participación activa). • Monitoreo de SFA disponible (control de calidad). • Conocimiento en profundidad de las tecnologías existentes. **Modelo de relación con el cliente** • El cliente no depende de un único proveedor (posesión del código, proveedores alternativos). • El valor añadido reside en el servicio. • Relación de larga duración (soporte, mantenimiento, versiones, etc.) • Gran diversidad de proyectos: comerciales, públicos, universitarios, etc.	**Ventajas para el cliente** • No paga licencias sobre componentes (bases de datos, servidores, sistemas operativos): - Ahorros de costes se pueden transferrir a mejorar el servicio. • Independencia del proveedor : - Acceso al código. - Existencia de proveedores alternativos con conocimientos sobre las tecnologías utilizadas. - Posibilidad de mejorar internamente (departamento de TI). • Longevidad de la solución en el tiempo (versiones). • Libertad de difusión (reproducción y redistribución). **Otros impactos** • ¿Qué obligaciones afectan a la redistribución? • ¿Exiten garantías sobre los componentes de SFA? • ¿El proveedor reutilizará el código?





El primer beneficio, y el más importante, para el consultor-integrador que utiliza SFA es **el amplio margen de libertad que dispone a la hora de crear aplicaciones para sus clientes, gracias a los derechos de copiar, modificar y redistribuir el software**[121] otorgados por las licencias libres.

Esta libertad se traduce en la posibilidad de:

- Integrar todo tipo de componentes de SFA para crear soluciones más complejas y/o adaptadas.
- Tener acceso al código fuente, para modificar y mantener el software.
- Eliminar partes innecesarias, para hacer un software más eficiente, menos pesado.
- Realizar todas las copias e instalaciones del software que uno quiera, u ofrecer los servicios del software en modo ASP (o Software as a Service).
- Reutilizar el software o sus módulos sin limitaciones (en uno o varios clientes).
- Ofrecer servicios de mantenimiento y evolución sin limitaciones.

Por otro lado, el proveedor gozará de una **gran independencia a la hora de proporcionar servicios de soporte y mantenimiento**, sin estar sujeto a obligaciones comerciales o de propiedad intelectual sobre el tipo de servicio que puede o debe ofrecer, o el tipo de modificaciones o mantenimiento (evolutivo) que puede realizar gracias al acceso al código fuente.

Asimismo, el SFA no obliga al integrador ni al cliente final a actualizar el software cada 3-4 años[122], otorgando un **ciclo de vida más largo** a sus aplicaciones. En la misma línea, el SFA suele funcionar de manera **más eficiente**, exigiendo menos recursos tecnológicos, lo que permite usar hardware con menos prestaciones y más económico. Esto ofrece una gran **estabilidad y continuidad tecnológica**.

Otro beneficio del SFA es la posibilidad de aprovechar **economías de escala**, consecuencia de no tener limitaciones de uso (y cobrar por usuario o CPU): una solución tendrá el mismo coste para un usuario que para 100. Por tanto, un integrador puede utilizar una misma aplicación o componente para un solo cliente o para 100 (sujeto solamente a lo que pueda acordar con sus clientes en cuanto a la confidencialidad y cesión de propiedad).

121 Los derechos de reproducción, transformación, distribución y comunicación pública, según la LPI (ver capítulo 7).
122 Por razones de ausencia de soporte, obligaciones contractuales o especificaciones de hardware.





6.2.2. Interrogantes

A pesar de sus múltiples ventajas, los profesionales del sector han planteado varias cuestiones relacionadas con los aspectos jurídicos que afectan al uso de SFA:

Preguntas

Estas son las cuestiones que suele plantearse un integrador de soluciones de SFA:

- ¿Cuáles son las obligaciones (relativas al código) que se imponen como consecuencia de usar SFA?
- ¿Cuál es el alcance del efecto llamado copyleft?
- ¿Se pueden usar componentes bajo una licencia u otra para un determinado cliente?
- ¿Se puede integrar un componente "X" con aquel otro "Y" y redistribuirlos juntos?
- ¿Se debe liberar (o publicar) el resultado de un trabajo de integración si se distribuye la solución a un cliente?
- ¿Cómo se puede o debe usar una marca de terceros, y cómo crear una marca propia para el producto?
- ¿Qué responsabilidades ante terceros tienen eventualmente los integradores y distribuidores de SFA?

Gran parte de las preguntas que se hace el integrador son muy similares a las que surgen en un proyecto de creación de SFA:

- Determinar y proteger la titularidad de los derechos del producto.
Cumplir con las condiciones de licencias sobre componentes.
- Publicar o no el código fuente.
- Cumplir con las obligaciones sobre el uso de marcas.

Esto es lógico si consideramos que la posición de un consultor-integrador es muy similar a la de un proyecto de creación de SFA: se desarrolla un producto a partir de uno o





varios programas de fuentes abiertas para distribuirlo al público o a un cliente particular. La principal diferencia es que el proyecto de creación se distribuye al público en general bajo una licencia libre, con el objetivo de obtener la mayor difusión posible, mientras que los resultados del trabajo del integrador suelen entregarse a una sola persona o entidad (o serie de personas o entidades, si se trata de un producto comercial) bajo un contrato comercial. Es decir, no será necesariamente liberado al público.

A continuación presentamos brevemente tres de los temas jurídicos más importantes que debe tener en cuenta el integrador:

1. El uso e integración de componentes SFA,

2. la selección de una licencia de distribución, y

3. las garantías y responsabilidades respecto del producto.

Después, comentaremos cómo se gestionan estos aspectos en el contrato de desarrollo y/o implantación.

6.2.3. El uso de componentes de software libre y de fuentes abiertas

En el caso de la integración y distribución de diferentes programas de software libre y de fuentes abiertas, el principal deber del proveedor es **respetar las condiciones de las licencias que existen sobre los componentes (en cuanto a su redistribución) y evitar conflictos (incompatibilidades) entre ellas.** Para ello, será importante conocer y entender la manera en que interactúan estos componentes y las consecuencias a nivel de compatibilidad de licencias[123].

Selección de componentes

El siguiente diagrama ilustra la arquitectura de una solución basada en SFA, incluyendo:

- Componentes estándares (Linux, Apache, PostgreSQL o MySQL, jBoss, etc.)

- Componentes más específicos para el programa en cuestión (las partes verdes)

- Nuevos desarrollos a medida según las necesidades del cliente (la parte violeta).

123 El concepto de compatibilidad de licencias es comentado en el Capítulo 3.





Arquitectura de una solución basada en SFA

A NIVEL DE SOFTWARE

Aplicaciones

eCommerce (Open for Busines)
- Catálogo de productos
- eCommerce
- Gestión de ordenes
- Shipment

- Buscador
- Lucene

- Módulos de
- impresión (Jasper)

Desarrollo a medida
- Gestión de la producción
- Facturación
- Cobro
- Gestión de la demanda
- Cuadro de Mando

Software Base / Infraestructura

Servidor Web: Apache	Base de Datos: PostgreSQL	Servidor de aplicaciones jBoss

Sistema Operativo

Linux

A NIVEL FÍSICO (HARDWARE)

www

Usuarios

Internet

Servidores de aplicaciónes y web

Servidores de bases de datos

El integrador deberá analizar la manera en que se incorporan los componentes de SFA, además de sus propios desarrollos. Así podrá identificar y eliminar (o sustituir) componentes que pueden generar conflictos de licencia a la hora distribuir el producto al cliente.





La integración de software: el impacto del copyleft

Este análisis es particularmente importante a la hora de usar componentes bajo licencia de tipo *copyleft*. En este caso se deberá asegurar la compatibilidad del resto de componentes con esta misma licencia, además de utilizarla para distribuir cualquier reproducción o modificación del software (junto con el código fuente). El uso de componentes bajo la GPL no solamente afecta al programa y sus modificaciones sinó también a las obras que se basan en él, algo que no es necesariamente fácil de interpretar y aplicar en la práctica. Será necesario estudiar cada caso, sobre todo teniendo en cuenta las diferentes maneras de interrelacionar los programas en la aplicación: enlaces estáticos, enlaces dinámicos, interfaces web (p.e. para *web-services*), etc.

Sin embargo, una aplicación que se ejecute sobre el sistema operativo GNU/ Linux o que use una base de datos MySQL (software bajo la GPL) no debe distribuirse necesariamente bajo la misma licencia con *copyleft*. Pero si se van a integrar *drivers*, librerías u otros módulos dentro del núcleo del sistema operativo, entonces se tendrá que considerar cuidadosamente cómo se realiza la integración y distribución del producto.

6.2.4. Licencias de distribución

En el caso de crear una aplicación basada en SFA para un cliente (o para el mercado en general) se deberá **evaluar y seleccionar la licencia de distribución más adecuada según los objetivos del producto o servicio y su modelo de negocio.**

De la misma manera que para un proyecto de software libre, las licencias sobre los componentes influencian directamente la selección de licencia de distribución:

• En el caso de integrar componentes bajo la GPL u otra licencia con copyleft fuerte, el integrador, generalmente, deberá distribuir su solución al cliente bajo esta misma licencia, proporcionándole el correspondiente código fuente. Esto no le impide cobrar por el producto o servicio. Lo que no puede hacer es cederlo al cliente bajo términos más restrictivos.

• Por el contrario, licencias más permisivas (como la BSD, MIT, etc.) le permiten integrar estos componentes en programas más complejos y distribuir el resultado final bajo otra licencia SFA compatible y hasta bajo una licencia no libre.

• Las licencias mixtas (o con copyleft suave, como la MPL, la CDDL, la OSL o la LGPL) permitirán al integrador cobrar por licencia (por una extensión propietaria agregada al producto original, por ejemplo), mientras que el SFA originario deberá distribuirse bajo la misma licencia.





Componentes copyleft o permisivos?

El consultor-integrador tendrá derecho a determinar las condiciones de la distribución del producto, siempre y cuando respete las exigencias establecidas en las licencias sobre los componentes que se explotan:

- Si lo que se quiere es vender licencias sobre el software, sólo será posible con componentes de SFA bajo licencias permisivas o, según el caso, con copyleft suave.

- Si el objeto del negocio es un servicio de implementación, soporte y mantenimiento, es posible el uso de componentes bajo cualquier tipo de licencia (la licencia no impacta el servicio comercial).

- Si el objetivo es vender parcialmente licencias basadas en una solución abierta (licencias sobre extensiones o personalizaciones verticales o sectoriales, etc.), la selección de los componentes implicados en el producto será más compleja.

6.2.5. Garantías y responsabilidades

El proveedor también deberá prestar una especial atención a las garantías que se den en relación con la distribución de software y la prestación de servicios basados en SFA, y las responsabilidades que se puedan derivar de ello.

Aunque sea una cuestión debatida, el Derecho español (aunque no necesariamente el americano) considera que el cliente / usuario debe beneficiarse de algunas garantías mínimas legales relativas al producto o servicio. El proveedor deberá responder ante determinados daños como la muerte y/o daños físicos causados por el software, así como en casos de dolo y de negligencia.[124]

Siguiendo la práctica de los EE.UU. (y la gran mayoría de las licencias estándares de software privativo), las licencias de SFA suelen incluir limitaciones de garantías y responsabilidades del tipo "este producto se entrega TAL CUAL, sin garantía alguna" (los disclaimers). Nuestras leyes pueden restringir la validez o alcance de estas limitaciones, según las circunstancias de cada caso. Por ejemplo, un tribunal reconocerá (generalmente) la validez de estos disclaimers cuando se trate de una relación equilibrada y/o negociada entre empresas o profesionales, pero no en el caso de una relación entre consumidores.

124 *Ver apartado vvv del capítulo 3. Por lo general, en cuanto a la muerte se refiere, no será el caso del software a no ser que se emplee en sistemas de alto riesgo.*

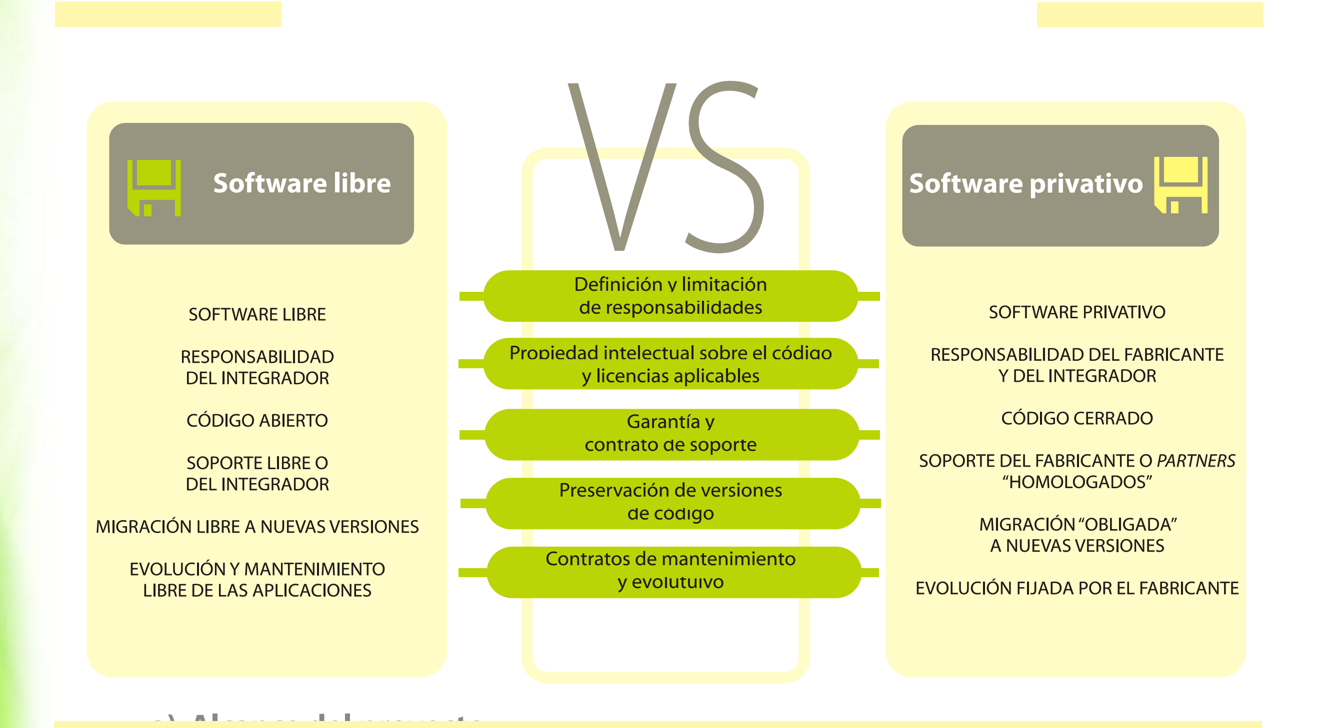




6.2.6. El contrato basado en SFA

El contrato entre el consultor-integrador y el usuario final es el último elemento de la cadena de comercialización de productos y servicios basados en SFA que nos queda por tratar. Sin perjuicio de las múltiples cuestiones que estos contratos originan, aludimos a cinco puntos importantes directamente relacionados con el SFA, ilustrados en el diagrama siguiente.

El contrato de implantación: software libre vs. sofware privativo



a) Alcance del proyecto.

La empresa integradora de SFA suele ser el único responsable ante el cliente, ya que será difícil exigir responsabilidades al proyecto o la comunidad del SFA en cuestión. Por esta razón será muy importante que el proveedor defina exactamente el ámbito del proyecto y el alcance del servicio prestado, detallando todas las tareas a realizar los productos a entregar. Sólo de esta forma el integrador podrá excluir o delimitar sus responsabilidades con respecto al SFA de terceros.

b) Propiedad Intelectual.

Como hemos visto, la comprensión de las licencias de fuentes abiertas y de su compatibilidad (en función de la arquitectura y métodos de integración del software) es importante de cara a asegurar y respetar la propiedad intelectual y evitar infracciones a derechos de terceros. El contrato deberá determinar el régimen de cesión de derechos (de acuerdo con las licencias preexistentes), el pago de licencias sobre el producto (en caso de que las haya) y las garantías respecto al cumplimiento de las mismas. Normalmente se excluyen de esta cesión los componentes de SFA, ya que el cliente recibe una licencia directa





del titular del software. Pero el contrato deberá contemplar los derechos sobre el software de nueva creación y las transformaciones de los componentes de SFA.

c) Garantías.

Aunque las licencias sobre los componentes de fuentes abiertas intentan excluir garantías y limitar responsabilidades, las soluciones entregadas suelen disponer de una garantía de funcionamiento durante un periodo de tiempo en función del equilibrio y el poder de negociación de las partes. En el caso de soluciones con módulos de SFA, el integrador puede quedar obligado a extender la garantía sin percibir ninguna remuneración económica a cambio. En estas circunstancias, será importante que el contrato determine claramente los componentes que están cubiertos por la garantía (y el coste económico para cubrirlos), el periodo de la misma y las condiciones para su ejecución.[125]

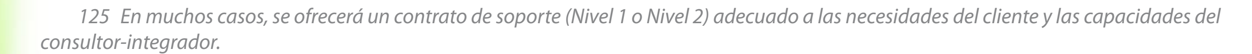
d) Preservación de fuentes.

El acceso al código fuente de una aplicación, así como su preservación, es causa de problemas tanto para el proveedor como para cliente. En los desarrollos en los que se utilizan componentes de SFA, muchos problemas desaparecen ya que el código fuente será accesible (y normalmente entregado) al usuario final. El contrato no suele especificar las condiciones en las cuales el cliente podrá tener el código fuente, sino simplemente quién será responsable de su preservación. Asimismo, se establecerán condiciones relacionadas con la migración a nuevas versiones (cuando se distribuyan con los proyectos originarios), con la corrección de errores o con la incorporación de nuevas funcionalidades.

e) Mantenimiento y evolución del software.

Una de las características más particulares de los productos basados en SFA es que el cliente es independiente de su proveedor. Con los derechos de explotación y el código fuente en mano puede cambiar con mayor facilidad de empresa integradora y mantener o evolucionar la solución implantada con otro proveedor en el caso de surja algún problema o éste deje de prestar sus servicios. Así, el mantenimiento y evolución del producto basado en SFA suele ser abierto y flexible.

6.3. EL USO DE SOFTWARE BAJO LA GPL: UN CHECKLIST



125 En muchos casos, se ofrecerá un contrato de soporte (Nivel 1 o Nivel 2) adecuado a las necesidades del cliente y las capacidades del consultor-integrador.





El siguiente cuadro muestra, a título de ejemplo, un checklist para integradores y vendedores de software que incluye SFA bajo la GPL, para asegurar que el usuario pueda beneficiarse de sus derechos y minimizar los riesgos de incumplimiento.

Antes de distribuir el software se debe:

• Asegurar que todos los titulares de los componentes incluidos en el software han cedido sus derechos y permiten distribuirlos junto con los componentes licenciados bajo GPL.

• Comprobar que los proveedores están obligados a rectificar en caso de incumplimiento de las condiciones o derechos establecidos.

• Verificar que dichos proveedores han proporcionado todos los materiales necesarios para poder cumplir con la GPL (código fuente, scripts, etc.).

• Constatar que los demás componentes de la aplicación, integrados con el SFA, tienen licencias compatibles con la GPL.

A la hora de distribuir el software

• Verificar que el paquete incluye una copia de cada licencia de los componentes del SFA.

• Revisar que se han mantenido o puesto los avisos de *copyright* de los titulares de los componentes bajo la GPL y cualquier otra licencia de SFA.

• Ver si el paquete incluye cualquier aviso legal adicional (Notice.txt, etc.) requerido por una u otra licencia de SFA.

• Examinar si el paquete incluye una copia del código fuente de cualquier software licenciado bajo la GPL (y sus obras derivadas) distribuido de forma binaria, o una oferta de proporcionar dicho código fuente al usuario.

• Confirmar que la versión del código fuente es la correcta y corresponde al código binario.

• Asegurar que se incluyen los *scripts* y otros elementos necesarios par volver a compilar e instalar el software a partir del código fuente.

NOTA: se aceptará el uso de un servidor de Internet para proporcionar el código fuente sólo si el software se distribuye a través de la red y el código está disponible en el mismo servidor que la distribución (o en uno distino, previo acuerdo de mantener el código a disposición de terceros).

6.4. CASOS DE ESTUDIO





CASO	OpenTrends Solucions y Sistemes S.L
La empresa	OpenTrends(*http://www.opentrends.net/web/sites/opentrends/ES/*) es una empresa de ingeniería de software que tiene como objetivo proporcionar soluciones y sistemas de información mediante la aplicación de tecnologías y metodologías de desarrollo de SFA. Su actividad se centra principalmente en la implantación de aplicaciones, entornos de desarrollo e infraestructura basados en SFA. Asimismo ha liberado su propio *framework* de desarrollo J2EE *openFrame*[126].
Cuestiones jurídicas	En gran medida, los proyectos de desarrollo e implantación de openTrends no plantean interrogantes legales, debido al alto nivel de coherencia en el desarrollo y los productos de cliente bajo la licencia GPL. Ello obliga, por ejemplo, a escoger componentes de SFA compatibles con esta licencia. Ha sido importante cuidar los siguientes temas: • El uso intensivo, por parte del personal, de software de fuentes abiertas, lo que ha generado la necesidad de integrar y distribuir soluciones a medida bajo la GPL. • La búsqueda de componentes de SFA maduros, estables y provenientes de fuentes sólidas desde una perspectiva tanto técnica como organizativa. • La gestión de la propiedad intelectual en casos de colaboración con otras empresas. • La definición de obligaciones, responsabilidades y garantías respecto a los componentes de SFA en los acuerdos con clientes finales.
Gestiones legales	Para hacer frente a estas cuestiones, openTrends toma las siguientes medidas: componentes de SFA. • Un benchmarking continuo de SFA disponible y de la evolución del sector. • El control y gestión de las licencias sobre componentes de terceros incorporados en las aplicaciones para clientes. • La firma de acuerdos de colaboración y subcontratación que especifican claramente la cesión o licencia de derechos sobre los desarrollos.

126 *http://www.opentrends.net/web/sites/opentrends/ES/Noticias/081106_openframe_release20.html*





	OpenTrends Solucions y Sistemes S.L
Gestiones legales	• La negociación y redacción de acuerdos con el cliente en función de los resultados esperados, en cuanto al alcance de cada proyecto, la propiedad intelectual, la gestión del código, soporte y garantías. • La búsqueda de soporte de segundo nivel, en los casos en que ese necesario, con el objetivo de respaldar los componentes de SFA incluidos en sus productos / servicios.
Lecciones	OpenTrends demuestra que una buena formación del personal y su orientación y/o motivación para usar SFA, una política coherente en cuanto a licencias y contratos con los clientes y una correcta gestión del software utilizado aumentan la eficiencia y eficacia en la ejecución de proyectos (además del ahorro de costes en términos de licencias que supone el uso de este tipo de software). Asimismo, el hecho de crear soluciones modulares y de mantener la titularidad del programa les permite reutilizar de manera eficiente sus trabajos, respetando en todo momento la confidencialidad de los desarrollos específicos para clientes. Axiom Tech, miembro del *Open Source Consortium* inglés[127], ofrece servicios de consultoría, desarrollo y soporte basado en SFA: la instalación y soporte Linux, sistemas 127 *http://www.opensourceconsortium.org/*





CASO	Axiom Tech Ltd
La empresa	de gestión de contenidos web, soluciones empresariales a medida, etc. Además, ofrece el servicio de correo/colaboración Zimbra como servicio remoto. Axiom Tech considera que los derechos concedidos por las licencias de SFA son una ventaja para la empresa que,
Aspectos jurídicos	esto suponga asumir riesgos mayores o distintos a los que surgen en un contexto privativo. La empresa resalta las siguientes implicaciones prácticas del régimen de licencias del SFA: • El uso de SFA optimizado que funciona sobre cualquier equipo, sin estar obligado a actualizarlo cada tres/cuatro años (por razones de falta de soporte, de especificaciones mínimas de hardware soportado u de obligaciones contractuales con el fabricante) permite ofrecer ahorros sustanciales en hardware a los clientes. • El libre uso de SFA, sin coste de licencia ni contratos de comercialización / distribución con los fabricantes, permite ofrecer una amplia gama de soluciones complejas (normalmente reservadas a grandes cuentas) a precios asequibles para todo tipo de cliente. • El ofrecimiento de “software as a service” bajo licencia SFA permite realizar una sola inversión en la infraestructura y ofrecerlo a un número ilimitado de usuarios. • La independencia del cliente (por no estar vinculado con un software sin código fuente) y sus economías de escala (por no pagar por usuario) cambia la dinámica de la venta de productos, enfocando el servicio al cliente, el soporte, la optimización de recursos y reducción de precios. • El hecho de que la empresa pueda retener la propiedad del desarrollo a medida (distribuyéndolo al cliente bajo licencia de SFA) tiene ventajas importantes a nivel de coste para el cliente (y beneficios de reusabilidad para Axiom Tech). Para aprovechar estas oportunidades, la compañía realiza un mínimo de formación al equipo de desarrollo y cuida la gestión de licencias sobre el software usado (controlando que no haya ningún software propietario en las soluciones entregadas). Las libertades del SFA (de usar el software cuántas veces se quiera sin pagar proporcionalmente al uso, ni estar obligados por contratos de exclusividad o distribución) permite realizar economías de escala tanto internas (en el propio uso y en





CASO	Axiom Tech Ltd
Lecciones	ofrecer servicios remotos) como externas (para el cliente). El acceso al código fuente y los derechos de modificación de las licencias de SFA son importantísimos para garantizar no solamente la libertad del cliente final, sino también del integrador, que debe proporcionarle adaptaciones específicas y soporte durante el ciclo de vida del software.

7.1. LAS FORMAS DE PROTECCIÓN JURÍDICA DEL SOFTWARE





CAPÍTULO 7

LOS DERECHOS DE AUTOR





Ficha resumen

OBJETIVO

Presentar, de manera general, como el software queda protegido por el Derecho de la propiedad intelectual (derechos de autor).

DESTINATARIOS

Todos los creadores, integradores y usuarios de SFA.

RESUMEN

El Derecho de la propiedad intelectual protege todo tipo de software, desde sistemas operativos hasta aplicaciones de usuario o de redes, junto con sus materiales preparatorios y documentación (diagramas, manuales, etc.).

Normalmente, los titulares del software son su autor (o grupo de autores) o la empresa en la cual trabaja el autor, sin perjuicio de aquellas obras que son resultado de una colaboración o están sujetas a la supervisión de un editor.

La ley concede en exclusiva:

- A los autores, ciertos derechos morales.
- A los titulares originarios, el derecho a realizar o autorizar a terceros la reproducción, transformación y distribución del software, además de su comunicación pública, por ejemplo mediante redes telemáticas.

Estos derechos están limitados por

- El tiempo (70 años a partir de la muerte del autor).
- Excepciones a favor de usuarios legítimos: la copia de seguridad y el derecho de reproducción y modificación del software para que su ejecución, y su descompilación (sujeta a condiciones que trataremos más adelante) a efectos de conseguir la interoperabilidad.

La exclusividad de los derechos permite a sus titulares autorizar y delimitar el acceso, uso y explotación del software por parte de terceros, a través de contratos o licencias.

Enlaces / referencias

En esta Guía:
- Capitulo 2, sobre licencias de Software de Fuentes Abiertas.
- Capítulo 3, sobre aspectos prácticos de las licencias de SFA,
- Capítulo 5, sobre la creación de SFA en proyectos.

En la Red:
Texto refundido de la Ley de propiedad intelectual: *http://noticias.juridicas.com/base_datos/Admin/rdleg1-1996.html*





Con el fin de fomentar la creación de nuevas obras (en este caso, de software), nuestro marco jurídico ha reservado una serie de derechos a los autores y titulares de las mismas, para que puedan controlar su explotación y defenderlas de la apropiación o uso ilegítimo por parte de terceros.

Sin perjuicio de otras figuras jurídicas que también otorgan protección al software (por ejemplo, la competencia desleal), l**os programas de ordenador pueden protegerse específicamente y de forma acumulativa bajo cuatro figuras legales o tipos de derecho**. En función de los distintos elementos que deseemos proteger, elegiremos una figura jurídica u otra y, en su caso, lo protegeremos acumulativamente con las figuras jurídicas que más nos interesen:

1. El código literal del programa y su documentación preparatoria y técnica se protegen expresamente por los derechos de autor.

2. Las funcionalidades y procesos de un programa pueden protegerse, aunque de manera controvertida, por la figura de la patente.

3. El derecho de marcas concede un derecho exclusivo y excluyente sobre el uso de una marca relativa a un software.

4. La figura del secreto industrial permite proteger la información confidencial concerniente a la creación del software, su diseño o sus modelos de implementación y de negocio que puedan tener un alto valor comercial.

Este capítulo y el siguiente se dedican a presentar y comentar cómo estas diferentes figuras jurídicas protegen, limitan y permiten la explotación del software.

7.2. LA PROTECCIÓN POR DERECHOS DE AUTOR

El software -incluida su documentación preparatoria y técnica- se considera una obra "intelectual" y está cubierto por los derechos de autor según la Ley de Propiedad Intelectual (LPI)[128]. Esta ley concede al titular orginario del software (normalmente el autor) el derecho a realizar y/o autorizar a terceros la copia, modificación y (re)distribución del programa[129]. Esta concesión exclusiva permite a los titulares del software autorizar y delimitar el acceso, uso y explotación del software por parte de terceros, a través de contratos o licencias.

Visión histórica

Inicialmente, el software no se comercializaba de manera independiente (pues se

128 Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el Texto Refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia.

129 En lenguaje jurídico, como veremos más abajo, la "reproducción", "transformación" y "distribución" del software, así como su "comunicación pública", por ejemplo mediante redes telemáticas. Para facilitar la lectura, utilizaremos con cautela los términos más usuales "copiar", "modificar" y "(re)distribuir".





distribuía con el *hardware*) y su protección se confundía con la del bien comercializado: el ordenador. Era difícil realizar copias o utilizarlo fuera del ordenador original, por lo que no existía una preocupación por su protección de manera específica e independiente a la máquina que se vendía.

Sin embargo, con el tiempo y los avances tecnológicos, el software obtuvo cada vez más relevancia, y su progresiva independencia respecto al hardware lo hizo fácilmente copiable. Para salvaguardar las inversiones necesarias para su creación (considerado como un trabajo intelectual estimable y valioso) se consideró que los derechos de autor (*copyright* en Derecho anglosajón) eran figuras ideales para conseguir una protección frente a la copia y uso ilícito.

La protección del Derecho de la propiedad intelectual ofrece varias ventajas frente a otras figuras jurídicas:

1. Es automática: los derechos de autor sobre el software nacen por el mero hecho de la creación original del mismo. Es decir, no es necesario inscribir el software en ningún registro público (como así ocurre con las marcas o patentes) para que se le otorgue la protección por derechos de autor.

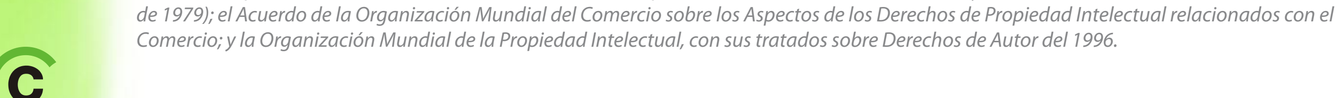
2. Es económica: al no hacer falta realizar solicitudes ni registrar el software, tampoco se deben pagar las correspondientes tasas y honorarios de registro o agencia para obtener la protección.

3.Es internacional: la protección nacional se extiende automáticamente a casi todo el mundo en virtud de tratados internacionales como el de Berna, el ADPIC y los acuerdos OMPI[130].

7.3. EL OBJETO DE PROTECCIÓN Y SUS TITULARES

Para entender el alcance de la protección del software por los derechos de autor, es necesario determinar qué es lo que se protege y quién tiene los derechos.

7.3.1. El software como obra sujeta de protección



130 Respectivamente, el Convenio de la Unión de Berna para la Protección de Obras Literarias y Artísticas de 1886 (con una ultima revisión de 1979); el Acuerdo de la Organización Mundial del Comercio sobre los Aspectos de los Derechos de Propiedad Intelectual relacionados con el Comercio; y la Organización Mundial de la Propiedad Intelectual, con sus tratados sobre Derechos de Autor del 1996.





La LPI incluye expresamente los "programas de ordenador" en la lista, no exhaustiva, de obras protegidas. Esta ley los define de la siguiente manera:

> "toda secuencia de instrucciones o indicaciones destinadas a ser utilizadas, directa o indirectamente, en un sistema informático para realizar una función o una tarea o para obtener un resultado determinado, cualquiera que fuere su forma de expresión y fijación (…) La expresión programas de ordenador comprenderá también su documentación preparatoria." (Artículo 96)

Notemos que esta protección se aplicará a cualquier forma de expresión de un programa de ordenador y su documentación preparatoria, pues se entiende que quedan protegidas todas las fases de elaboración del programa desde el momento en que hay una descripción del mismo, en forma gráfica (diagrama de flujos o de tipo UML) o verbal (grabada), suficientemente detallada para determinar un conjunto de instrucciones.

Los contenidos y la estructura de una base de datos no se consideran programas de ordenador. Sin embargo gozarán de una protección general por derechos de autor si son obras originales (como cualquier obra original) o una protección específica establecida en la LPI para bases de datos[131]).

Obras protegidas

La definición legal incluye cualquier tipo de programa:

1. Sistemas operativos, programas estándares de uso general y programas desarrollados a medida.

2. Compiladores, librerías/bibliotecas y otros componentes de software, scripts, servlets, Java Beans, stored procederes, etc.

3. Motores de bases de datos.

4. Entornos de desarrollo (IDEs, p.e. Eclipse del Proyecto Eclipse, el JDK de Sun, .NET de Microsoft) y de ejecución (runtime engines).

5. Servidores de aplicaciones y web, etc.

6. Instrucciones incorporadas en los chips (microcódigo y firmware).

7 .Programas que soportan las redes de telecomunicaciones (routers, switches, servidores, firewalls, etc.).

131 Conocida dentro de la LPI española, al igual que en otras europeas, como "sui generis".





Respecto a estos programas, los derechos de autor cubren:

1. El código fuente y objeto del software programado.

2. Las imágenes, iconos, gráficos, etc. del programa o incluidos en el programa.

3. Los guiones o scripts de compilación e instalación.

4. La documentación preparatoria: diseños y diagramas, especificaciones y listados, modelos de datos.

5. Cualquier documentación técnica y manual de instalación o de usuario.

6. Versiones sucesivas de un programa.

7. Cualquier modificación de estas obras (obras derivadas).

8. De manera separada y especial, la estructura y contenido de bases de datos.

Como condición para tener esta protección un programa debe ser original, en el sentido de ser una creación intelectual propia de su autor. Se entiende, pues, que se ha optado por un criterio de bajo nivel (en relación con los criterios aplicables a otros tipos de obras): que el software sea meramente el resultado de un esfuerzo personalizado, es decir, básicamente que no esté copiado.

Es importante destacar que la protección por los derechos de autor no afecta a las ideas y principios en los que se basan cualquiera de los elementos de un programa, incluidos los que sirven de fundamento a sus interfaces.

Esta limitación permite, entre otras cosas:

• La creación de software con funcionalidades similares a otro, por ejemplo a partir de una ingeniería inversa[132] o especificación independiente[133]. Pensemos en programas para la edición de imágenes como Adobe Photoshop, CorelDraw y The Gimp.

• El desarrollo de software que puede interoperar con otro a través de APIs u otra forma de interfaz.

132 Esto sólo será legal si se hace dentro de las posibilidades que permite la Ley de Propiedad Intelectual.

133 Comenzando a reprogramar de cero, por el método que se conoce como "clean room development", para evitar el riesgo de infracción por plagio.





7.3.2. Los autores y titulares del software

Con dos excepciones, **el titular originario de los derechos sobre el software coincidirá, en principio, con el autor o grupo de autores que lo crearon.** Este grupo incluirá a todas las personas que participaron en su creación, como el analista, el jefe de proyecto, el diseñador gráfico, etc.

Aviso de titularidad en las cabeceras

Existe una presunción jurídica según la cual el autor y/o titular del software es la persona identificada en la firma (por ejemplo, en el aviso de derechos de autor o "*copyright notice*") o el que la publica. El aviso de titularidad (que no autoría) del software deberá cambiar en función de las posibles cesiones de derechos que se formalicen, posteriormente y de forma escalonada, en el tiempo.

Si miramos una cabecera de un fichero de SFA, veremos la mención de titularidad:

© 2007 Mozilla Foundation
Original authors: John Doe, Mary Jones, etc.
Modifications: © Peter Smith 2008 - Author of modifications: Peter Smith

Las dos excepciones en las que, desde un inicio, la titularidad de los derechos de explotación sobre el software difiere de su autor, considerado como una persona física, son:

1. El software creado bajo una relación laboral. Salvo pacto contrario, los derechos de explotación del software creado por trabajadores asalariados en el ejercicio de sus funciones o siguiendo las instrucciones del empresario, corresponden al empresario.[134]

2. El software que se crea como obra colectiva. Los derechos de explotación del software creado bajo la iniciativa y coordinación de un editor, resultado del conjunto de varias aportaciones de diferentes autores, pero en el que "la reunión de las aportaciones personales de los diferentes autores se funde en una creación única y autónoma para la cual haya sido concebida, sin que sea posible atribuir separadamente a cualquiera de ellos un derecho sobre el conjunto del software", se entenderán que son titularidad de ese editor.

134 En el mismo contexto, los derechos del software creado para una empresa por subcontratas, colaboradores en régimen de autónomos o becarios, serán de estas personas y no de la empresa, a no ser que se interprete algo distinto de la relación o se pacte lo contrario.





Identificación del titular

Identificar al titular de los derechos del software es importante por varias razones. El titular es la persona que:

• Deberá ser identificada en el aviso de *copyright* ("© año, titular"),

• tiene el derecho a establecer las condiciones de distribución (la licencia) y de exigir el cumplimiento de las mismas, y

• tiene los derechos morales (siendo una persona física).

Con independencia de la relación laboral, en el contexto del SFA, son pocas las veces en las que una sola persona es el único autor de un software. Normalmente, los programas han sido creados por equipos de personas (grupos de investigación o compañeros de trabajo, no todos asalariados) o en colaboración con personas casi desconocidas, pero en el que las contribuciones son claramente diferenciables. En estos casos, no siempre puede decirse que todos los que participaron en el desarrollo son autores en una situación de igualdad, y no siempre será fácil determinar la proporción de los derechos de explotación que le corresponde a cada uno.

En estos casos de autoría múltiple, existen otras figuras jurídicas que identifican al autor y titular originario de los derechos de explotación aunque, desafortunadamente, no siempre se corresponde con la realidad de un proyecto de SFA. Éstas son la **obra colectiva** y la **obra en colaboración.**

• **La obra colectiva** (ya definida anteriormente) es la creada por iniciativa de un coordinador que la edita y divulga bajo su nombre. Los derechos de explotación serán únicamente del coordinador-editor.

• **La obra en colaboración** es el resultado unitario de la colaboración de varios autores, en el que es posible disgregar las aportaciones de cada uno y explotarlas por separado. Los autores son titulares, de forma conjunta y a porcentajes, de la obra total y se requiere el consentimiento de todos para su explotación.





Obras colectivas y en colaboración de SFA

En el caso de software coordinado dentro de un proyecto Apache, se podría considerar a la Fundación Apache como editor y titular de los derechos, dependiendo de su rol en la coordinación del proyecto. Ello sin perjuicio de que la Fundación exija que cada contribuidor firme una cesión de derechos a la Fundación, para mayor seguridad legal.

Otro ejemplo podría ser un grupo de investigación universitaria, en el cual el investigador principal – empleado de la universidad – inicie el proyecto y coordine la labor de sus compañeros de trabajo, y el resultado sea publicado por la misma universidad. En este caso, ésta podría argumentar que es titular de los derechos del software como obra colectiva.

Un ejemplo de una obra en colaboración es el resultado del trabajo de un grupo de desarrolladores donde no hay ninguna coordinación ni divulgación específica por parte de una persona o entidad central, sino una publicación conjunta de los resultados por parte de todos los miembros del grupo. En este caso, el aviso de copyright sería:

© 2008 Juan Gómez, Julie Blanco, Pedro Jiménez

7.3.3.Tipos de obras

Actualmente, tanto el software en general como el SFA no suelen ser una obra de nueva creación en su totalidad, sino la composición o integración de varios programas preexistentes. Para estos casos la LPI establece dos figuras relacionadas entre sí:

- **La obra compuesta** es la formada por varios programas independientes preexistentes, incorporados mediante una copia y/o modificación.[135]

- **La obra derivada** es la que se obtiene a partir de otra anterior (preexistente) por su modificación o adaptación. En el caso del software, se define específicamente como una "traducción, adaptación y cualquier otra modificación en su forma de la que se derive una obra diferente".

En ambos casos, los derechos de estas obras pertenecerán al autor de las mismas, quien necesitará la correspondiente autorización de los titulares de cada componente para copiar y/o modificar los programas que va a utilizar. Esta autorización se consigue a través de una licencia.

135 Un factor que distingue la obra compuesta de la obra colectiva o en colaboración es que su desarrollo no queda sujeto a una coordinación entre los contribuidores y/o con la persona considerada por la ley como titular de la obra final (lo que no implica falta de comunicación entre ellos). Ciertos autores lo consideran una forma particular de obra derivada.





Tipos de obra en el SFA

Cualquiera que haya creado un software a partir de varios componentes - librerías, iconos, servidores de aplicaciones, bases de datos, etc. - reconocerá la obra compuesta. Así, esta figura es mucho más cercana a la realidad de la mayoría de los proyectos de SFA (que incorporan componentes de terceros) y, sobretodo, de las soluciones y aplicaciones completas creadas para clientes.

En la práctica, la mayoría de los programas de SFA son fruto de los esfuerzos de varios autores y han sido creados a partir de varios componentes. Por lo tanto, generalmente, son obras en colaboración o compuestas. Pero también pueden ser productos empresariales, como el Zope CMS (de la Zope Corp.) o MySQL (de MySQL AB, ahora parte de Sun Microsystems Inc.) o, en casos contados, obras colectivas.

La ventaja de las licencias reconocidas de SFA es que permiten, en todo los casos, esta reutilización de componentes y la creación de obras compuestas o derivadas, sin necesidad de obtener el permiso expreso de los titulares.

7.4. LOS DERECHOS DE LOS TITULARES, SUS LÍMITES Y CESIONES

En este apartado veremos los derechos que la ley reserva a los titulares, cuáles son sus límites y cómo pueden cederse a terceros.

7.4.1. Los derechos de autor del software

Los derechos o facultades del titular de un programa incluyen los derechos morales (si el autor es persona física) y los derechos patrimoniales de explotación.

Los **derechos morales** están vinculados con una persona física (el autor). Se refieren, entre otros, a los derechos de autoría o paternidad (la obligación de identificar siempre al autor) y de integridad (la posibilidad de impedir la modificación de una obra para proteger la reputación del autor). Incluyen también el derecho a determinar la divulgación de la obra y la forma en qué se realiza, y el derecho a retirarla del mercado, previa indemnización para aquellos terceros que hayan adquirido derechos de explotación de la misma.

Los derechos morales son irrenunciables e intransmisibles: acompañan al autor durante su vida y, bajo determinadas circunstancias, incluso después.





Derechos morales

Las licencias de SFA implementan un mecanismo que hace respetar los derechos morales (aunque, paradójicamente, fueron creadas bajo el régimen del copyright anglosajón, que no reconoce estos derechos): exigen el reconocimiento del titular de los derechos y obligan a mantener el aviso de copyright, así como a identificar las modificaciones realizadas En algún caso incluso se obliga a redistribuir cualquier obra derivada como un parche separado, para evitar el riesgo de mermar la calidad del software original y la correspondiente reputación de su autor.

En cuanto a los derechos de explotación, aunque de manera general hablamos de "copiar, modificar y (re)distribuir" el software, la ley se refiere específicamente a:

1. La reproducción del programa: la copia total o parcial del programa, por cualquier medio y bajo cualquier forma, para uso personal o no, ya fuere permanente o transitoria. La LPI incluye cualquier reproducción realizada durante la carga, presentación, ejecución, transmisión o almacenamiento de un programa (incluso en RAM).

2. La transformación: consiste en la traducción, adaptación, arreglo o cualquier otra transformación del programa, y la reproducción de los resultados de dichos actos. El resultado es la creación de una "obra derivada".

3. La distribución del programa: se trata de la transmisión de la obra en un medio físico a un tercero, mediante la venta, el alquiler o cualquier otra forma (por ejemplo vender o distribuir una copia del software en un CD/DVD).

4. La comunicación pública[136]: acto por el cual una pluralidad de personas puede tener acceso a la obra sin previa distribución de ejemplares. Incluye la "puesta a disposición de terceros de la obra por procedimientos alámbricos o inalámbricos de tal forma que cualquier persona pueda acceder a ellas desde el lugar y en el momento que elija". Una manera de decir "colgar en Internet".

Estos derechos de explotación pueden cederse a terceros inter vivos o por mortis causa, de manera conjunta o separada, y sujeto a las condiciones que determine el titular (ver el apartado 4.3 de este capítulo, más adelante).

Sin esta cesión (licencia), cualquier explotación de un programa, ya sea su copia, modificación o distribución, estará restringida o prohibida por defecto.

REPETIDO

– el contexto y el texto es diferente – pero la práctica del snippet es tán común que lo pondríamos.

136 Aunque este derecho no esté recogido entre los derechos específicos de explotación de los programas de ordenador,debe entenderse implícitamente aplicable a los mismos.





7.4.2. Limites a los derechos de autor

Los derechos de autor ofrecen una protección limitada en el tiempo, con el objetivo de permitir la reutilización de la obra una vez el titular ya ha obtenido los beneficios derivados de su explotación. Este limite se ciñe, generalmente, a la vida del autor más 70 años, después de los cuales el programa será de dominio público y utilizable por todos sin condiciones[137]. Cuando hablamos de software, esta limitación en el tiempo tiene escasa relevancia.

La LPI establece otros límites específicos para los derechos de explotación del software. Existen excepciones a favor de los usuarios legítimos que afectan a los derechos exclusivos, con el fin de garantizar que el software se utiliza de acuerdo con su finalidad inicial.

Excepciones

Los usuarios legítimos del software pueden:

• Copiarlo y modificarlo cuando sea necesario para permitir su uso de acuerdo con su finalidad (por ejemplo, la corrección de errores), siempre que contractualmente no se disponga lo contrario.

• Realizar una copia de seguridad (cuando sea necesario y si el titular no le ha provisto de una) .

• Estudiar su funcionamiento, para entender las ideas y principios que subyacen al programa.

• Realizar una descompilación de partes del software con fines de interoperatibilidad, siempre que el titular no haya facilitado la información necesaria para ello (código fuente, APIs, etc.). La información recogida sólo será utilizada para este objetivo específico y no para crear programas similares (tampoco puede transmitirse a terceros).

• Ninguna de estas excepcions podrá aplicarse si perjudican injustificadamente al autor original.

En el caso del software (ya sea de SFA o de otro tipo), estos límites serán relevantes, por ejemplo, a la hora de crear un programa que interactúe o que intercambie información con software propietario.

137 Pero siempre con respeto a los derechos morales, que son indisponibles e irrenunciables.





7.4.3. La cesión de derechos de autor: la licencia de software

Un aspecto fundamental para el software –dado que, a menudo, el usuario no es el autor– es que el titular de los derechos de explotación puede cederlos a un tercero, por contrato de cesión o licencia.

El art. 43 LPI prevé:

> *"Los derechos de explotación de la obra pueden transmitirse por actos inter vivos, quedando limitada la cesión al derecho o derechos cedidos, a las modalidades de explotación expresamente previstas y al tiempo y ámbito territorial que se determinen".*

Este contrato o licencia puede imponer condiciones al ejercicio de los derechos cedidos, ya sean libremente pactadas entre las partes (un contrato negociado) o determinadas unilateralmente por el titular (como por ejemplo, las licencias estándares de software de tipo ALUF[138], incluidas las licencias de SFA).

En particular, **los derechos pueden ser cedidos de manera exclusiva o no exclusiva:**

• Una **cesión exclusiva** atribuye al cesionario (licenciatario) la facultad de explotar la obra con exclusión de otras personas, incluido el propio titular (cedente). Asimismo, salvo que se pacte lo contrario, permite al cesionario otorgar (posteriores) autorizaciones no exclusivas a terceros y lo legitima para emprender acciones legales para defender los derechos de autor.

•En cambio, una **cesión no exclusiva** reserva al titular originario la posibilidad de volver a ceder derechos sobre el software a terceros.

Si no se especifica lo contrario, las licencias se presumen no exclusivas, intransferibles y personales. Y se entiende que la cesión se lleva a cabo únicamente para satisfacer las necesidades del usuario. Por esta razón es importante detallar adecuadamente el alcance de la cesión de los derechos.

Las licencias de SFA son no exclusivas: el titular retiene sus derechos de licenciar el software a otros, bajo la misma u otra licencia. Asimismo, salvo autorización específica, es el único legitimado a defenderse de las infracciones de derechos de autor e incumplimientos de la licencia.

138 Pero siempre con respeto a los derechos morales, que son indisponibles e irrenunciables.





Finalmente, "*toda cesión deberá formalizarse por escrito*" (Art. 45 LPI), de manera física (en papel) o electrónica (el contrato *on-line, browser-wrap, clic-wrap o shrink wrap*, como comentamos en el Capítulo 3 de esta Guía).

7.5. LAS MEDIDAS DE PROTECCIÓN JURÍDICA

Además de la posibilidad de utilizar el símbolo © y de acceder al Registro de la Propiedad Intelectual[139], la normativa española establece varias medidas para proteger el software de un uso ilegítimo. Se prevén acciones administrativas, civiles y penales, dentro de las cuales incluso se podrán solicitar medidas cautelares[140]:

- **Las acciones civiles, además de imponer el cese de la acción, también permiten solicitar una indemnización por los daños materiales y morales causados**. Entre otros, la tutela civil prohíbe las copias ilegales, su posesión para fines comerciales, o la puesta en circulación o posesión de medios específicamente destinados a suprimir o neutralizar los dispositivos técnicos de protección del software (como por ejemplo "*cracks*", "*keygens*" o números de serie).

- El Código Penal castiga la **reproducción, transformación, distribución y comunicación pública de software con ánimo de lucro, en perjuicio de terceros y sin autorización de los correspondientes titulares de las obras protegidas por derechos de autor.** Asimismo, prohíbe la importación, exportación y almacenamiento de copias ilegales, y la fabricación de medios específicamente destinados a facilitar la supresión o neutralización de dispositivos técnicos de protección (de tipo DRM[141] como el CSS[142], las claves de acceso, sistemas de registro, etc.).

7.6. LOS DERECHOS DE AUTOR Y EL SFA

Es importante entender el efecto del Derecho de propiedad intelectual y la filosofía que subyace a esta regulación para comprender la genialidad de las licencias de SFA.

Actualmente, el principio fundamental de este Derecho es ofrecer al titular de una obra mecanismos para poder beneficiarse de su explotación, tanto si se realiza personalmente como a través de terceros: los derechos exclusivos. Además, los titulares pueden establecer

139 *Acuerdo de Licencia de Usuario Final (o EULA en inglés, End User License Agreement).*
140 *La inscripción no es obligatoria, pero establece una presunción de titularidad.*
141 *Digital Rights Management: medidas tecnológicas de protección de los derechos de autor. Comentado en Internet en http://es.wikipedia.org/wiki/Gest*
142 *Content Scrambling System: sistema de cifrado de contenidos, des-cifrado por el DeCSS. Descrito en Internet en http://es.wikipedia.org/wiki/Content_Scrambling_System*





condiciones a la explotación por parte de terceros, recogidas en la licencia: el pago de un precio o regalías, condiciones en cuanto a copias, modificaciones y redistribución.

El SFA cambia radicalmente esta filosofía, respetando en todo momento los derechos de autor. Basándose en la titularidad y las facultades exclusivas garantizadas por ley, **el principio del software libre y de fuentes abiertas es ceder los derechos a todo el mundo, bajo ninguna o muy pocas condiciones, para que cualquiera pueda explotar el programa libremente.** En el Capítulo 3 de esta Guía explicamos cómo las licencias de SFA implementan esta filosofía.

Consideramos que se trata de una reingeniería admirable del marco jurídico, invirtiendo la posición legal por defecto, para “permitir casi todo”, en vez de “restringir casi todo”.





CAPÍTULO 8

OTRAS NORMAS APLICABLES





Ficha resumen

OBJETIVO

Mostar la protección delsoftware mediante los Derechos de propiedad industrial (patentes y marcas) y de secreto industrial.

DESTINATARIOS

Todos los creadores, integradores y usuarios de SFA.

RESUMEN

El software puede protegerse jurídicamente por las figuras de la patente, la marca y el secreto industrial.

• Una **patente** puede (aunque este es un tema controvertido) dar a su titular derechos excluyentes sobre las funcionalidades y procesos implementados en un software, lo que impedirá cualquier tipo de reingeniería o que estos puedan volverse a implementar en otro contexto. La protección por patente tiene carácter territorial, y sus criterios varian en función del país o zona. En Europa, el "software en sí" está excluido de la patentabilidad, lo que no impide que la Oficina Europea de Patentes y el resto de oficinas nacionales otorguen patentes de "software con un efecto técnico".

• Una **marca** sirve para identificar un producto o servicio y la empresa que los crea y/o comercializa. Nuestra legislación prohibe que terceros utilicen una marca registrada sobre determinados productos y servicios. De esta manera, cualquiera que desee relacionar su software con una marca deberá contar con la suficiente autorización del titular.

• El derecho de la **competencia desleal** nos protege, entre otras cosas, de la revelación y uso desleal de información confidencial, lo que puede incluir no solamente el código fuente de un software, sino también la información técnica y económica de una empresa de SFA que explote su software.

Enlaces / referencias

En la Guía:
Capítulo 2, sobre licencias de SFA (licencias de patentes y marcas)

Capítulo 5, sobre proyectos de creación de SFA (selección de marca, protección de secretos, etc.)

En la Red:

Oficina Europea de Patentes – *www.oep.org*

Oficina Española de Patentes y Marcas – *www.oepm.es*





8.1. EL DERECHO DE LA PROPIEDAD INDUSTRIAL

El objetivo del Derecho de la propiedad industrial es el de proteger los resultados de la labor inventiva y creativa en un contexto industrial y económico (en contraste con la propiedad intelectual, que cubre obras artísticas, literarias y científicas, incluido el software). Para ello se han creado dos figuras jurídicas, **la patente y la marca**, que pueden aplicarse al software (aunque este un tema controvertido), tal y como comentamos a continuación:

• **Las funcionalidades de un software podrían protegerse mediante la figura de la patente**, siempre que se consideren una invención y cumpla las condiciones establecidas en la Ley de Patentes[143]. Esta protección es mayor que la otorgada por los derechos de autor y le da al inventor y/o titular el monopolio de explotación del software. Sin embargo, su aplicación al software es muy controvertida.

• El software (y los servicios sobre el mismo) suele ir asociado a un signo distintivo o "marca", como Mozilla®, Suse® o Apache®. **Según el Derecho de marcas, el titular podrá utilizar su marca de forma exclusiva y excluyente** para todas aquellas categorías de productos y servicios para la que haya sido registrada[144]. Así pues, la ley protegerá el valor y posición competitiva de un proyecto o una empresa basada en SFA, pero puede que esto entre en conflicto con las libertades concedidas por sus licencias.

8.2. EL DERECHO DE LAS PATENTES

El principal objetivo de las patentes es el de fomentar las invenciones. Para ello reserva a los inventores el derecho a ser los únicos que exploten sus propias ideas. A cambio, éstos deben publicar los detalles de su invención, para que otros puedan conocerla y crear nuevos productos en base a ésta y/o explotarla con el debido consentimiento (pagando una licencia).

143 Ley 11/1986, de 20 de marzo, de Patentes de Invención y Modelos de utilidad.

144 Ley 17/2001, de 7 de diciembre, de Marcas para la marca nacional española; Reglamento (CE) nº 40/94 del Consejo, de 20 de diciembre de 1993, sobre la marca comunitaria.





8.2.1. La patente

La patente es un derecho de propiedad industrial que se concede en relación con una invención. Es un derecho negativo, ya que faculta al titular para impedir que terceros produzcan, comercialicen, usen, distribuyan o vendan productos que incorporan la invención sin su autorización.

La protección de la patente está limitada:

Al territorio donde se ha registrado y concedido. Será necesario solicitar el registro de patente a cada una de las oficinas correspondientes a los distintos territorios de interés[145] y **en el tiempo**, normalmente para un periodo de 20 años, después del cual cualquiera puede explotar la invención. Se trata de un tiempo muy inferior al establecido por los derechos de autor pero, en el caso del software, continua siendo innecesariamente largo, dada la vida útil de cualquier programa.

El titular de una patente puede denunciar cualquier actividad infractora e impedir el uso del proceso o producto patentado, aunque esta explotación sea involuntaria. Los creadores o intermediarios en la cadena de suministro de software pueden infringir una patente al desarrollar, distribuir o comercializar un software infractor, o al integrarlo en otros productos. En el caso de un usuario final, la infracción puede venir del mero hecho de usarlo.

8.2.2. Invenciones patentables y obtención de patentes

Las invenciones patentables son productos o procedimientos (métodos para hacer algo) <u>nuevos</u>, resultado de una <u>actividad inventiva</u> y con una <u>aplicación industrial.</u>

145 Discutiblemente, la distribución en línea de un producto infractor – un software, por ejemplo – pone a su distribuidor en riesgo de ser demandado en los países en los que se pueda acceder al producto, aunque no haya patente concedida (o válida) en el país de origen del distribuidor.





Requisitos para solicitar una patente

• **Novedad:** la invención no forma parte del *Estado de la Técnica* (esto afecta, prácticamente a cualquier información ya existente y publicada en el mundo en el momento de la solicitud).

• **Actividad inventiva:** la invención no debe ser obvia para alguien que es experto en la materia, en base a todo lo que ya existe y es conocido.

• **Aplicación industrial:** la invención debe tener aplicación en cualquier sector de la economía (incluidas las telecomunicaciones y la informática).

Cuando hablamos de software, debemos añadir un cuarto requisito relacionad con su "*efecto técnico*", que viene determinado por alguno/s de los requisitos anteriores.

Quedan excluidos de la patentabilidad, entre otros, ciertos resultados de la labor humana como, por ejemplo, los descubrimientos científicos y métodos matemáticos, los planos, los métodos para realizar actividades mentales, juegos o actividades económico-comerciales, **los programas de ordenador** y la presentación de la información, todos ellos considerados por sí mismos.

La exclusión del software se limita al "programa de ordenador en sí mismo", lo que ha generado un debate entorno a si un programa puede considerarse o no algo más que una serie de instrucciones informáticas. Trataremos este tema más adelante.

Obtención de patentes

Para obtener una patente hay que presentar una solicitud ante la correspondiente oficina de patentes, sea nacional o regional (la Oficina Europea de Patentes)[146], cada una con sus propias tasas y trámites.

La solicitud debe describir la invención y su contexto, con diagramas, planos, etc. En especial, debe incluir las "reivindicaciones", a partir de las cuales se protegerán jurídicamente los productos y/o procesos que se describen. Se suele contratar a expertos en la materia (compuesto de equipos mixtos de agentes de patente, ingenieros y abogados) para redactar la patente y asegurar las máximas oportunidades de éxito.

146 Se puede solicitar una patente para más de un país, a través de un procedimiento internacional gestionado por la OMPI que permite, con un solo documento, solicitar la protección de la patente en varios Estados. Notemos que la Unión Europea tiene planes controvertidos para crear una patente comunitaria, así como un tribunal único a nivel europeo para las patentes europeas.





Una vez hecha la solicitud, la oficina de patentes la revisa e investiga los criterios de patentabilidad y, en caso de que se satisfagan (en su opinión), publica la solicitud para obtener comentarios u objeciones de terceros. Si no las hay, después de un periodo de tiempo, se concede la patente. Este proceso suele durar entre 18 meses y 3 años.

Explotación de la patente

Una vez concedida la patente por la oficina correspondiente, el inventor tiene el monopolio de explotación del producto o proceso protegido, con carácter retroactivo desde su solicitud. Puede explotar el producto él mismo o autorizar a terceros a hacerlo en su lugar, de manera exclusiva o no, a través de una licencia de patente. Asimismo, la patente es un título de propiedad, por lo que el inventor podrá ceder (vender) la patente a un tercero. Bajo ciertas condiciones, el inventor puede verse obligado a otorgar una licencia, precisamente para evitar que las invenciones, dada la concesión tan amplia y monopolista, queden sin explotar.

8.2.3. Las patentes de software

Como hemos visto anteriormente, los derechos de autor protegen el código y la documentación de un software, pero no las ideas y principios en que se basa. De esta manera es posible crear un nuevo software con funcionalidades similares a otro programa sin infringir los derechos de autor. Para evitar esto, la industria tradicional tiende a adoptar la figura de la "patente de software", con el objetivo de restringir el desarrollo de programas que ofrecen a los usuarios las mismas soluciones que su propio software.

Con la patente no sólo se protegerá el código y documentación de un programa, sino también cualquier otra implementación de los procesos o funciones patentadas, extendiendo la protección jurídica a las ideas que subyacen al software.

La protección del software por patentes

La mayoría de disputas jurídicas con más repercusión económica relacionadas con el software no tocan el tema de derechos de autor, sino el de patentes. *Eolas vs. Microsoft, NTP vs. RIM (Blackberry), Symantec vs. Microsoft*, han sido algunos casos en Estados Unidos; los casos de *Macrossan, Aerotel o Haliburton* en Inglaterra[147]. Una conclusión preliminar llevaría a pensar que el software quedará mejor protegido con una patente.

147 *Ver en http://www.ipo.gov.uk/patent/p-decisionmaking/p-law/p-law-notice/p-law-notice-subjectmatter.htm*





Aunque la patentabilidad del software es controvertida, no deja de ser una práctica permitida tanto a nivel de registro como de disputas ante los tribunales:

- Los que están a favor de esta práctica, normalmente empresas de la industria del software privativo (con portafolios de patentes)[148], argumentan que la mayoría de las invenciones actuales incluyen alguna forma de software y que el software tiene un efecto técnico y aplicación industrial. Por tanto, hay que permitir que se patente.

- Los que se oponen a la patentabilidad, la comunidad del SFA en particular, argumentan que el software es una expresión de reglas matemáticas, sin afectar a las "fuerzas de la naturaleza", y está considerado como un producto intelectual.

Efecto técnico

El marco jurídico de las patentes en Europa excluye de la patentabilidad a los programas "en sí". Sin entrar en detalles, la Oficina Europea de Patentes (OEP) considera como "software en sí" el que no incluye o no se asocia con algún efecto técnico. Generalizando (en Derecho español y en un entorno europeo), podríamos decir que **los programas patentables serían aquellos que tienen un "efecto técnico" más allá de la mera interacción entre el programa y el ordenador**[149].

El efecto técnico

Es interesante resaltar que la OEP ha presentado una casuística que centra su atención en lo que es o no es un efecto técnico. Más complicado es explicar qué es este "efecto técnico", p entendiendo que se refiere a la relación del software con una máquina o dispositivo de hardware.

Sin embargo esta interpretación no es más que una definición predominantemente subjetiva (existiendo diferencias, incluso, entre lo que opinan los tribunales sobre la cuestión) elaborada fundamentalmente a nivel de la OEP para que, con ciertas sutilezas y en ocasiones excepcionales, se permita proteger el software caso por caso, mediante una patente más o menos fuerte.

Por tanto, a pesar de la exclusión del software "en sí", en los últimos 10 años, la OEP y la Oficina Española de Patentes y Marcas han concedido patentes de software alegando un efecto técnico, incluso en el caso de software que implementa métodos de negocio, también excluido por el Convenio. [150]

148 Y patent trolls : empresas que tienen patentes como únicos activos y viven de derechos de licencia.

149 Siempre y cuando se hayan reunido también los requisitos de patentabilidad: es decir, la novedad, actividad inventiva y aplicación industrial.

150 Debemos mencionar que, a nivel español, las disputas sobre patentes de software son escasas y normalmente, los tribunales españoles siguen el mismo criterio marcado por las salas de recursos de la Oficina Europea de Patentes.





Patentes de software a nivel internacional

En 2005, se intentó incorporar la patentabilidad del software en el marco jurídico de la Unión Europea, con la fallida y rechazada Propuesta de Directiva sobre las Invenciones Implementadas por Ordenador. En los EE.UU., y algunas otras jurisdicciones como Japón y Australia, se permite conceder patentes sobre "cualquier cosa inventiva y útil", incluso el software, a pesar de existir un gran debate entorno a la utilidad y legalidad de su patentabilidad.

En términos generales, se estima que se han concedido más de medio millón de patentes sobre software a nivel mundial.

8.2.4. Los riesgos de la patente de software

A pesar de todos los debates y dudas que existen entorno a las patentes de software, cualquier persona implicada en la creación o explotación de SFA debe ser consciente de las problemáticas que pueden surgir en este aspecto.

Efectos nocivos

1. Si existe una patente sobre el proceso implementado por el software, entonces el titular de la patente puede impedir que se pueda desarrollar y difundir cualquier otro programa que tenga las mismas funcionalidades.

2. La patentabilidad del software impide, en cierta manera, la creación de interfaces para la interoperabilidad de programas, algo que está específicamente permitido bajo el régimen jurídico de derechos de autor.

Estas son algunas de las causas de la oposición a las patentes de software de la comunidad SFA, ya que requerirían una investigación previa, exhaustiva y costosa del estado del arte y el software registrado bajo una patente[151], figura que además impide las libertades propugnadas por el movimiento FOSS.

No se puede hacer una reingeniería de un proceso patentado o reescribir una aplicación desde cero. Básicamente hay que obtener una licencia (ya sea pagando su precio u obteniéndola a cambio de otras patentes: licencias cruzadas de portafolio o cartera de patentes) o tratar de invalidar ante los tribunales la patente (lo cual, en la práctica, tiene unos costes bastantes elevados)). En algunos casos, la solución es contratar un seguro.

151 Normalmente, de sus descripciones y de lo que se reivindica en los documentos de patente. No es una tarea trivial el poder identificar de manera fácil en qué consisten exactamente las invenciones.





Patentes y SFA

La posibilidad de patentar software supone un serio problema para la comunidad de SFA: sus proyectos no tienen patentes para intercambiar, no tienen fondos para defenderse en los tribunales y no tienen el tiempo ni los recursos para dedicarse a la investigación de patentes (con la excepción de las grandes empresas involucradas en el movimiento, como Novell, Sun o IBM). La mera amenaza de demandas puede ser un freno para un proyecto, dado el coste que implicaría defenderse en un tribunal.

Si bien es cierto que los proyectos de SFA no son un objetivo de los titulares de patentes, si lo podrían ser las empresas usuarias de SFA: grandes empresas, administraciones públicas, etc.

Es importante recordar que las licencias del SFA, como casi cualquier licencia de software, no incluyen ninguna garantía respecto a la infracción de derechos de patente de terceros.

El riesgo de patentes no es diferente para software privativo o libre. Lo que sí está claro es que la disponibilidad del código fuente simplifica enormemente la tarea de averiguar y descubrir posibles infracciones, mostrando las pruebas y sirviendo el pleito en bandeja. Por esta razón, la posición del SFA puede ser más delicada y peligrosa.

Las licencias de SFA más modernas (desde la MPL hasta la reciente GPLv3) incluyen cláusulas de cesión de derechos y de defensa ante eventuales patentes, con el objetivo de crear un espacio libre de ellas e impedir su uso en el entorno del SFA (*ver el Capítulo 3 de esta Guía*).

8.3. EL DERECHO DE LAS MARCAS

Una marca sirve para identificar un programa en el mercado (como si de cualquier otro producto se tratara), y así como los servicios asociados al mismo. Es decir, utilizar una marca es una manera de garantizar al usuario el origen e, indirectamente, la calidad del correspondiente software.





8.3.1. La obtención de marcas

Una marca es un signo distintivo (denominativo, gráfico o mixto) que permite identificar y distinguir los productos o servicios de una empresa. Para obtener protección jurídica, debe básicamente cumplir dos requisitos:

1. Tener carácter distintivo; y

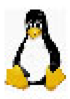
2. ser susceptible de representación gráfica.

Una marca puede incluir palabras, logos, letras, números, colores, dibujos, formas tridimensionales, signos percibidos por los sentidos (sonoros, olfativos, gustativos y táctiles) o una combinación de éstos.

Marcas denominativas y gráficas

Algunos ejemplos de marcas denominativas son "Microsoft", "Red Hat", "IBM", "Linux", "Mozilla" o "Windows", que, además de identificar determinados productos o servicios también identifican a las empresas que hay detrás de ellos.

Los proyectos de SFA suelen identificarse con iconos – marcas gráficas - muy reconocibles. Pero no todos serán marcas registradas (aunque igualmente están protegidos por derechos de autor, por ser obras gráficas).














Para que una marca esté protegida jurídicamente, será necesario registrarla ante una Oficina de Marcas (con algunas excepciones[152]).. La protección se solicita en función de una (o más) categorías de productos y/o servicios, organizadas en una clasificación armonizada[153].

152 También se pueden obtener derechos por usar el signo por primera vez y de forma efectiva para designar un producto, hasta tener cierta notoriedad en el mercado.

153 Llamada el Nomenclátor Internacional o "Clasificación de Niza".





Existen dos tipos de prohibición para registrar una marca:

• **Absolutas**, p.e. signos contrarios al orden público o a los principios de moralidad comúnmente aceptados; y

• **relativas**, para proteger derechos anteriores (de personas que ya tienen una marca protegida)no se puede registrar signos idénticos o similares a marcas existentes para las mismas categorías de productos o servicios.

La marca estará protegida en el territorio de la oficina nacional en cuestión, de la Unión Europea en caso de solicitar una marca comunitaria, o en los países indicados en la solicitud en caso de usar el sistema internacional de registro (sistema del Protocolo o Arreglo de Madrid).

8.3.2. Derechos y obligaciones de una marca y su cesión

El registro de la marca confiere a su titular dos derechos:

1. El de usar la marca para distinguir sus productos y/o servicios en el mercado.

2. El de prohibir el uso de un signo idéntico o similar por parte de terceros.

De esta manera, se pretende evitar que otros puedan aprovecharse de la reputación de una marca concreta para vender sus propios productos y la protege de la imitación, la copia o la falsificación.

Violación de una marca

Son ejemplos de violación de una marca: el uso de un signo idéntico o similar al de una marca existente para un mismo tipo de producto (Lindows / Windows), aplicar una marca a un producto sin autorización del titular (usar "Java" en la distribución del software sin permiso de Sun Microsystems, Inc.), o la venta de software pirateado (con la marca sobre el envoltorio).

Los derechos duran 10 años y, si la marca se sigue usando, pueden renovarse (pagando la tasa correspondiente) de forma indefinida por periodos también de 10 años.

Pero ser el titular de una marca también implica asumir ciertas obligaciones. En primer lugar ésta debe ser utilizada de forma efectiva en el mercado. Después se debe proteger del uso indebido de terceros sin autorización, para mantener su carácter distintivo y el vínculo con determinados productos. En caso contrario la marca se vulgarizará, perdiendo el derecho a reclamar su uso exclusivo.





Los derechos de uso de la marca pueden cederse por licencia (por ejemplo, IBM los cede a sus socios comerciales, para fines de publicidad o marketing). Esta cesión puede ser exclusiva (por ejemplo, para un distribuidor exclusivo de un producto en un mercado) o no, y bajo las condiciones pactadas en la licencia de la marca.

Políticas de marca y licencias

Varios proyectos SFA(como por ejemplo Mozilla, osCommerce o MySQL) publican una "política de marca", para ofrecer licencias limitadas sin que sea necesario solicitar una autorización particular.3 Otros proyectos incluyen pactos relativos a la marca en la licencia (en la Academic Free License, por ejemplo, o la prohibición del uso de la marca "Apache" incluida en la licencia Apache 1.1).

8.3.3. Marcas y software de fuentes abiertas

Los creadores de SFA se preocupan de forma especial por mantener su reputación como proveedores de software de calidad, identificar cuál es su software y cuándo ha sido modificado por terceros. Una marca aporta una capa de protección adicional, orientado al uso del software en el mercado. Los distribuidores y otros intermediarios pueden, con autorización del titular, usar una marca para identificar el SFA que distribuyen o implementan, y aprovecharse su reputación.

El uso de marcas en el SFA

Las marcas intervienen en varios ámbitos del SFA:

1. Cuando se elige un signo para identificar y distribuir el software.

2. Cuando se redistribuye. Aunque el software sea de uso libre, la marca no lo es: hay que respetar las obligaciones y la política de marca establecidas por el titular, en cuanto a su uso en materiales publicitarios, interfaces gráficas y otros elementos del software. RedHat especifica, por ejemplo, que: "no permite ni autoriza el uso de sus marcas de manera que pueda causar confusión en el mercado por implicar el patrocino por o la asociación con RedHat".

3. Cuando se modifica y redistribuye. Es importante considerar que los derechos de modificación no se aplican a la marca. Los titulares suelen prohibir su uso para promocionar o identificar software modificado.

4. Cuando se crean programas compatibles con o para un software determinado. Se hará referencia a la marca de dicho software en la promoción, publicidad o distribución del programa: p.e., en frases como "compatible con software XXX", "usa software YYY", programado en ZZZ".





Los titulares de una marca deben controlar el uso de la misma para mantener su protección legal, de lo contrario puede ocurrir lo que se llama "vulgarización", caso en el que la marca pierde su carácter distintivo. Para evitar su vulgarización, las políticas de marca de los proyectos de SFA suelen ser más exigentes (menos libres) que sus licencias. [154]

Linux® en Australia

En Australia, un tribunal denegó la protección legal del signo "Linux". Su titular, Linus Torvalds, no tomó las medidas necesarias para su protección jurídica, y el tribunal consideró que demasiadas personas habían usado el término en relación con sus propios productos (la mayoría, derivados del kernel o núcleo del sistema operativo GNU/Linux) para que la marca pudiera mantener su distintividad en ese territorio.

8.4. EL DERECHO DE LOS SECRETOS INDUSTRIALES

Podría parecer que la naturaleza abierta y el acceso al código fuente del SFA (además de la publicación de muchos de sus documentos técnicos y planes de proyectos) se contradice con el concepto de "secreto industrial", al menos, tal y como se concibe en el movimiento del software de fuentes abiertas.

Sin embargo, al liberar los programas bajo licencias libres, el negocio de las empresas que se basan en éste suele estar, precisamente, en lo que no se libera. Los beneficios pueden provenir de la cesión del uso de una marca o de permitir el acceso y empleo de información específica de valor añadido. Así pues, queda claro que las libertades propugnadas por la filosofía del SFA no implican, necesariamente, que deba explicarse o revelarse absolutamente toda la información o conocimiento sobre el software.

8.4.1. Protección jurídica del secreto

Es un secreto industrial toda aquella información que tiene un valor económico independiente (ya sea actual o potencial) y que se ha tratado de forma confidencial. Así, los secretos industriales contendrán aquella información que dan al empresario una ventaja competitiva respecto a sus competidores. Puede incluir documentos, fórmulas, modelos, compilaciones, programas, mecanismos, métodos, técnicas o procesos, etc.

154 Por ejemplo, en http://www.mozilla.org/foundation/trademarks/policy.html o http://www.mysql.com/about/legal/trademark.html





Secretos en el SFA

Cuando hablamos de SFA, esta información puede incluir todo lo relacionado con el *know-how* de una empresa vinculado con el software(documentos, metodologías, especificaciones, etc.), técnicas de consultoría, optimización del producto, metodologías de implantación en clientes o procesos de venta, etc. así como sus propios planes de negocio.

La protección jurídica del secreto industrial tiene el fin de impedir la divulgación de esta información con valor económico por parte de aquellos que la conocen– por haber participado en la creación y definición de la estrategia de negocio de una empresa, producto o servicio, por ejemplo.

Esta información confidencial no sólo está protegida por la legislación laboral y penal, sino también por la normativa que regula la competencia desleal, que castiga la violación de los secretos industriales y su explotación de manera ilegítima por parte de terceros. Existen penas y sanciones que se traducen en indemnizaciones por daños y perjuicios.

Obligaciones de secreto

Las obligaciones de mantener el secreto forman parte de la relación laboral entre empleado y empresario (a menudo reforzadas por pactos explícitos de confidencialidad) y pueden establecerse mediante contrato (entre cliente/proveedor o consultor, investigador/universidad, etc.), a menudo conocido por su abreviación inglesa "NDA" (*Non Disclosure Agreement*).





8.4.2. Secretos industriales y software de fuentes abiertas

Los secretos industriales afectan a los proyectos de SFA en dos aspectos:

A) La protección de la propia información confidencial: una empresa que quiere establecer una línea de negocio basada en SFA estará interesada en determinar lo que es información confidencial dentro de la información manejada internamente en la empresa (y compartida entre consultores, socios, clientes, etc.) y tomar las medidas necesarias para protegerla. Esta protección se realiza a través de medidas tecnológicas (controles de acceso) y legales (firmando los adecuados acuerdos de confidencialidad).

B) El riesgo de violar los derechos sobre información de terceros:

• Un desarrollador que está sujeto a obligaciones de confidencialidad podría revelar secretos sobre metodologías, procesos, documentación técnica y hasta código fuente.

• Un empleado u otras personas que tengan acceso legítimo a cierta información confidencial (como por ejemplo un cliente o los empleados de éste) podría revelar know-how de una empresa sobre un SFA a través de su publicación en Internet (en el portal del proyecto, un blog, foro o chat público).

SCO c. IBM

SCO alegaba, en su querella contra IBM, que ciertos desarrolladores de esta compañía habían revelado secretos relacionados con el código de su versión de UNIX, que SCO había compartido en un proyecto de investigación. Según SCO , algunos empleados de IBM que participaron en este proyecto pasaron información confidencial a los que trabajaban en la unidad “Linux” de esta empresa.



